

مركز البحوث

نمذجة البيانات في قواعد البيانات والتحويل بين النماذج

1011001011011010010100101001001010001001

1101100101101101001010010100100101000100

تأليف

حليم حبيب حنا

بسم الله الرحمن الرحيم



مركز البحوث

نمذجة البيانات في قواعد البيانات والتحويل بين النماذج

تأليف

حليم حبيب حنا

عضو هيئة التدريب بمعهد الإدارة العامة - سابقًا

عضو هيئة التدريس بمعهد تعليم الفنادق الامريكية

فلوريدا - بالولايات المتحدة الأمريكية

١٤٢٣هـ - ٢٠٠٢م

بطاقة الفهرسة

ⓒ **معهد الإدارة العامة ، ١٤٢٢هـ**

فهرسة مكتبة الملك فهد الوطنية أثناء النشر

حنا ، حليم حبيب

نمذجة البيانات فى قواعد البيانات والتحويل بين النماذج - الرياض

٤٨٠ ص ؛ ١٦,٥ × ٢٣,٥ سم

ردمــــك :

٥ - ٨٦ - ١٤ - ٩٩٦٠

١ - قواعد البيانات أ - العنوان

ديوى ٠٠٥,٧٤ ٢٢/٠٩٢٠

رقم الإيداع : ٢٢/٠٩٢٠

ردمــــك : ٥-٨٦-١٤-٩٩٦٠

المحتويات

الموضوع	الصفحة

تابع - المحتويات

الموضوع	الصفحة

تابع - المحتويات

الموضوع	الصفحة

تابع - المحتويات

تابع - المحتويات

المقدمة :

تعتبر قواعد البيانات الإدارة الرئيسية لحل المشاكل الملحة فى مجال تشغيل البيانات. وقد هيمن على السوق ثلاثة نماذج رئيسية فى قواعد البيانات: النماذج الهرمية، والشبكية، والعلاقية على التوالى. وقد تميز كل من النموذج الهرمى والشبكى باستخدام الروابط الواضحة بين السجلات، حيث يتم التبحر والتعامل معها فى مراحل مختلفة على عكس النموذج العلاقى الذى لا يسمح بالتبحر خلال السجلات، ولكنه يتطلب تعريف طريقة التعامل مع السجلات فى بداية التعامل فقط. ثمة فرق آخر بين النموذجين الهرمى والشبكى من جهة والنموذج العلاقى من جهة أخرى وهو أن لغات الاستعلام الأساسية للنوعين الأولين تعتبر لغات من المستوى الأدنى فى حين يقابلها فى النموذج العلاقى لغات استعلام من المستوى العالى.

وعلى الرغم من نجاح نظام قاعدة البيانات العلاقية فى السنوات القليلة الماضية فى الوصول إلى مستوى الأداء المطلوب للتعامل مع التشغيل البيئى على نطاق واسع ؛ إلا أن الاتجاه إلى استخدام نموذج البيانات الشيئى الموجه مستمر الآن.

ومن ناحية أخرى فإن التطبيقات المختلفة المعتمدة على مقدرة النظم العلاقية فى نمذجة البيانات محدودة للغاية ، وتقع فى مجالات التصميم بالحاسب (CAD) والتصنيع بمساعدة الحاسب (AM) والبرمجة الهندسية بمساعدة الحاسب (CASE) والميكنة المكتبية، لكن نظم إدارة قواعد البيانات العلاقية سوف تستمر فى الهيمنة لسنوات عديدة مقبلة نتيجة للقوة التنفيذية الفعالة التى جعلتها المعيار الرئيسى لتطبيقات تشغيل البيانات.

وقد تم تقديم المفاهيم الأساسية والأسس العلمية لهذه النماذج المختلفة. وعرض لمفاهيم نموذج كينونة – علاقة ER والتطورات التى ألحقت به بوصفه نموذجًا للبيانات الدلالية التى تعبر عن العالم الحقيقى، وأداةً لتصميم قواعد البيانات المختلفة. ويساعد الإلمام بهذه الأسس فى تصميم قواعد البيانات التقليدية (التبحرية – والعلاقية) والشيئية الموجهة بشكل دقيق وخالٍ من تضارب البيانات.

هناك حالات كثيرة يكون فيها تحويل قواعد البيانات من نموذج إلى آخر أساسى وضرورى فيما بين نماذج قواعد البيانات التقليدية؛ لذلك كان حتمياً التوجه نحو وضع الأسس العلمية والعملية المبنية على المفاهيم الدقيقة للتحويل فيما بين هذه النماذج. وهذه الأسس تتضمن القواعد المنطقية للتحويل فيما بين النماذج التقليدية. وكذلك التحويل من نموذج قاعدة البيانات الشيئية الموجهة إلى قاعدة بيانات علاقية مع تقديم أمثلة مختلفة لتوضيح المنهاج المستخدم لتنفيذ نظم التحويل المقترحة بين نماذج قواعد البيانات. إلى جانب تقديم الخطوات الأساسية لتصميم نظام لأداة تصميم قاعدة البيانات الشيئية الموجهة Object-Oriented Database Tool System. وقد تم تحليل وتصميم هذا النظام بناء على تقنية النمذجة الشيئية Object Modeling Technique (OMT). وتعد تقنية النمذجة الشيئية من أفضل أدوات البرمجة الهندسية بمساعدة الحاسب CASE فى تحليل وتصميم قواعد البيانات الشيئية الموجهة. وتعتبر أداة تصميم قواعد البيانات الشيئية الموجهة من المتطلبات الضرورية لإنشاء قاعدة البيانات الشيئية الموجهة من خلال نمذجة أتوماتيكية مبنية على الأسس العلمية للنموذج الشيئى الموجه والتى تعد تطبيقًا علمياً لذلك النموذج.

الفصل الأول
مقدمة فى قواعد البيانات



مقدمة :

سوف يتم التطرق فى هذا الفصل لعدد من الموضوعات ذات الصلة بأساسيات قواعد البيانات ونماذجها المختلفة. ويعد هذا بمثابة مدخل يمكن القارئ من التعرف على مفاهيم قواعد البيانات الأساسية؛ ومن ثم يسهل عليه التعرف على موضوعات الكتاب وفهمها بشكل أكثر سهولة وتيسيراً. وفيما يلى عرض مختصر لما سيتم طرحه فى هذا الفصل من موضوعات:

نبذة تاريخية عن نظم قواعد البيانات :

يمكن تصنيف نظم قواعد البيانات المختلفة تاريخياً إلى قسمين من النظم. أولهما نظم قواعد البيانات التقليدية والآخر هو نظم قواعد البيانات الشيئية الموجهة. وتنقسم نظم قواعد البيانات التقليدية بدورها إلى نظم قواعد البيانات التبحرية والتى تتضمن كلاً من نظم قواعد البيانات الهرمية والشبكية والأخرى تشمل قواعد البيانات العلاقية. وسوف يتم التطرق فى الفصول القادمة إلى نماذج هذه النظم بأشكالها المختلفة ومعرفة الفوارق بينها؛ مما يتيح إمكانية التحويل فيما بين هذه النظم بشكل علمى دقيق.

المفاهيم الأساسية لقواعد البيانات :

سوف يتم التركيز فى هذا الجزء على الملفات وما يتعلق بها. حيث تشكل الملفات القالب الأساسى لقواعد البيانات ؛ ومن ثم يكون حتمياً معرفة تكوينها.

محتويات نظام قواعد البيانات :

يتضمن نظام قاعدة البيانات أربعة محتويات رئيسية هى:

البيانات Data ، والأجهزة Hardware ، والبرمجيات Software ، والمستخدمون Users.

قاعدة البيانات :

تتشكل قاعدة البيانات من مجموعة من الملفات المترابطة فيما بينها ، وتحتوى هذه الملفات على أنواع مختلفة من البيانات سوف يتم شرحها مع ذكر مدى الاستفادة من استخدام قواعد البيانات.

نظم إدارة قواعد البيانات :

تمثل نظم إدارة قواعد البيانات توضيحًا لمفهوم النظام البرمجى الذى يسمح بالتخزين الملائم للبيانات داخل قواعد البيانات. ويتم تنفيذ العمليات الضرورية عليها طبقًا لمتطلبات المستفيدين. وهناك العديد من وظائف نظم إدارة قواعد البيانات التى سيتم التطرق إليها.

البناء المعمارى لنظم إدارة قواعد البيانات :

قد تم تجهيز البناء المعمارى لنظم إدارة قواعد البيانات بواسطة المعهد القومى الأمريكى للمقاييس. وقد وضعت النظم القياسية لنظم قواعد البيانات فى أوائل السبعينات.

إمكانيات نظم قواعد البيانات :

سوف يتم توضيح إمكانيات نظم إدارة قواعد البيانات الشائعة والتى تميزها عن غيرها من النظم البرمجية الأخرى.

كفاءة التعامل مع الملفات :

توفر نظم إدارة قواعد البيانات المستخدمة نموذجًا يسمح برؤية البيانات فى مستويات عديدة فى شكل ملفات مدمجة؛ مما يسهل التعامل معها بشكل كفء وبسيط.

لغات الاستعلام :

سوف تتم الإشارة إلى لغة الاستعلام المبنية على أساس النموذج العلاقى ، حيث تتطلب هذه اللغة تفاصيل أقل من تلك الموجودة فى نماذج البيانات الأخرى سواء التبحرية منها أو الشيئية الموجهة .

نظم أساس الشىء :

يتم استخدام مفهوم أساس الشىء لتوصيف نوع من النظم البرمجية بإمكانيات نظم إدارة قواعد البيانات.

نبذة تاريخية عن نظم قواعد البيانات :

بدأ حقل إدارة قواعد البيانات فى أواخر الستينيات ؛ وكان ذلك استجابة لمتطلبات التحكم فى زيادة حجم البيانات الضخمة الناتجة عن أعمال المؤسسات المالية. وقد أسست نظم إدارة قواعد البيانات Database Management systems (DBMSs) فى البداية على الرسم الموجه graphically-oriented لنماذج تخزين البيانات. وقد بنيت نماذج البيانات الشبكية Network والهرمية Hierarchical على أساس الرسم الموجه وعرفت بالنماذج التبحرية Navigational Models.

قدمت هذه النظم أمثالاً من خلال نظم إدارة قواعد البيانات للنموذج الشبكى عرفت باسم تخزين البيانات المتكامل Integrated Data Store (IDS) لشركة Honeywell. كما قدمت مثالاً للنموذج الهرمى عرف باسم نظم إدارة المعلومات Information Management system (IMS) لشركة IBM.(1)

ثم ظهر فى أوائل السبعينيات مفهوم لقواعد البيانات العلاقية التى بنيت على الأساس الجدولى الموجه tabular-oriented ، وقد تم اقتراح النموذج العلاقى على أساس رياضى mathematical لتحليل ونمذجة البيانات واستقلاليتها. وقد قدمت أسلوباً للعنونة الزائدة للبيانات بالإضافة إلى التقدير الجيد لهياكل قواعد البيانات بأسلوب منهجى.

تمكن النموذج العلاقى من التعامل مع قيود السلامة integrity constraints ومخطط الأمن security schema وتوزيع البيانات distribution of data وتكرارها replication التى لم تكن محددة ومحللة من قبل بشكل دقيق جداً.

وعلى الرغم من ذلك لم تنتشر نظم إدارة قواعد البيانات العلاقية حتى أوائل الثمانينيات. وفى أواخر الثمانينيات كانت معظم نظم إدارة قواعد البيانات المنتشرة إما تبحرية أو علاقية. وكان العديد من الاقتراحات الخاصة بنمذجة قواعد البيانات البديلة منتشر فى ذلك الوقت منها نموذج البيانات الدلالية Semantic Data Model (SDM). ويرجع الدافع وراء تطور نموذج البيانات الدلالية تشابهه مع الاتجاه الشيئى object orientation لنموذج العالم. فى نمذجة البيانات الشيئية الموجهة object-oriented كل

كينونة entity يتم تمثيلها بمجموعة أشياء objects وعمليات operations مرتبطة بها. ويتركب الشيء من جزء من أشياء/أشياء فرعية تعبر عن العلاقة بين الأشياء. ويطلق على العمليات اسم الطرق (البرامج) methods، وقد يكمن عملها فى استدعاء عمليات خاصة بأشياء أخرى أو تغيير حالة أشياء معينة. ودخلت نظم إدارة قواعد البيانات الشيئية الموجهة عالم الأسواق كمنتج فعال فى منتصف الثمانينات. ونتيجة لتباعد الفترات الزمنية بين ظهور كل نموذج وآخر فإن العديد من النظم المعتمدة على هذا النموذج يتم تطبيقها خلال تلك الفترة. وقد أدى تباعد الفترات الزمنية إلى تأخر تنفيذ معظم التطبيقات المرتبطة بنظم قواعد البيانات العلاقية وانتشار التطبيقات المرتبطة بالنظم التبحرية؛ مما يتطلب صيانة هذه النظم الآن والتى يصعب هدمها لارتفاع تكاليف تصميمها ودقة عملها. وقد حصلت نظم قواعد البيانات العلاقية حديثًا على انتشار يفوق نظم قواعد البيانات الأخرى لقلة تكاليف تصميم التطبيقات الخاصة بها وسهولتها ، فى حين بدأت فى الظهور تطبيقات قليلة مرتبطة بالنظم الشيئية الموجهة[٢].

المفاهيم الأساسية لقواعد البيانات :

وقد بدا واضحًا فى العقد الأخير ، والسنوات الأخيرة على وجه الخصوص ، استعمال العديد من الأشخاص والمؤسسات لنظم البيانات وبشكل يومى ؛ لذا من الضرورى طرح المفاهيم الأساسية لقواعد البيانات والتى تمثل الملفات Files المحتوى الأساسى لها.

الملف File :

يتكون الملف من جزأين ، أحدهما يمثل هيكل الملف (مخطط الملف schema) يحتوى على مجموعة من الحقول fields (الأعمدة columns أو الخصائص attribute) والتى يتم توصيفها كقوالب لتحوى بداخلها البيانات. فى حين يمثل الجزء الآخر السجلات records (الصفوف rows أو القيم المرتبة tuples) التى تحوى بداخلها القيم الفعلية لهذه الحقول. وينبغى التمييز بين مصطلحين هما: البيانات Data والمعلومات Information وأحيانًا كثيرة يخلط بينهما فى المعنى. وتشير البيانات إلى القيم المخزنة فعليًا فى قاعدة البيانات ويجب أن تكون متكاملة Integrated وموضع مشاركة Shared. فى حين

يمكن الإشارة إلى المعلومات على أنها معنى هذه القيم كما يتم فهمها بواسطة المستخدم؛ ومن ثم يمكن الاستفادة منها. ويمثل الشكل رقم (١-١) ملف "الموظفين" EMPLOYEES الذى يحتوى على مجموعة من أسماء الحقول هى: المعرف ID ، الاسم Name، العنوان Address، رقم التليفون Tel_No والجنس Sex. وهى تعطينا معلومات عن أنواع الأشياء Objects الموجودة داخل ذلك الملف. فى حين توجد البيانات الفعلية فى بقية الصفوف. وكل صف يحتوى على معلومات مترابطة تعبر عن كينونة Entity خاصة لموظف معين والتى يطلق أحيانًا عليها سجل record. ويحتوى كل عمود على قيم خاصة للحقل (الخاصية) مرتبطة مع ذلك العمود بحيث تمثل كل قيمة داخل العمود معنى ذلك الحقل (الخاصية). ويعتبر الملف file جدولاً table ذا بعدين للأشياء objects ، أحدهما: يمثل الصفوف (السجلات) rows والآخر يمثل الأعمدة (الحقول أو الخصائص) columns. ومع ذلك يشار إلى اصطلاح ملف للتخزين الخارجى للبيانات فى حين يشار إلى اصطلاح "جدول" للتخزين داخل الذاكرة الرئيسية وإن كان كلاهما مرادفًا للآخر. كذلك ينبغى التمييز بين كلمتى مخطط schema وواقعة instance. فكلمة مخطط schema تستخدم لتوصيف الخصائص attributes والتى يطلق عليها هيكل الملف (الجدول)، وأنواع القيم values المسموح بها لكل خاصية ، وعلاقات الربط re- lationships بين الجداول (الملفات) فى قاعدة البيانات . وتعنى كلمة وقائع instances القيم المرتبطة بقاعدة البيانات وهى تصف محتويات الملفات (الجداول). على سبيل المثال تمثل القيم فى الشكل رقم (١-١) وقائع instances لملف (جدول) "الموظفين" EM- PLOYEES.

فى نظم قواعد البيانات يتطلب الأمر توصيف نوع كل خاصية attribute فى المخطط schema، على سبيل المثال نوع العنوان Address قد يكون أحرفًا (20) char وقد يكون نوع المرتب salary رقميًا Number وهكذا.

شكل رقم (١ –١) ملف «الموظفين» EMPLOYEES Flie

الجنس	رقم الهاتف	العنوان	الاسم	الهوية
M	3602703	Cairo	أكرم صلاح	11
M	8346666	Dammam	أحمد سالم	22
F	2425555	Cairo	هالة فؤاد	33
M	8337777	Dammam	محمد سليمان	44
F	6239999	Jaddah	عائشة عمر	55
M	5603516	Khartoum	جعفر الماحى	44

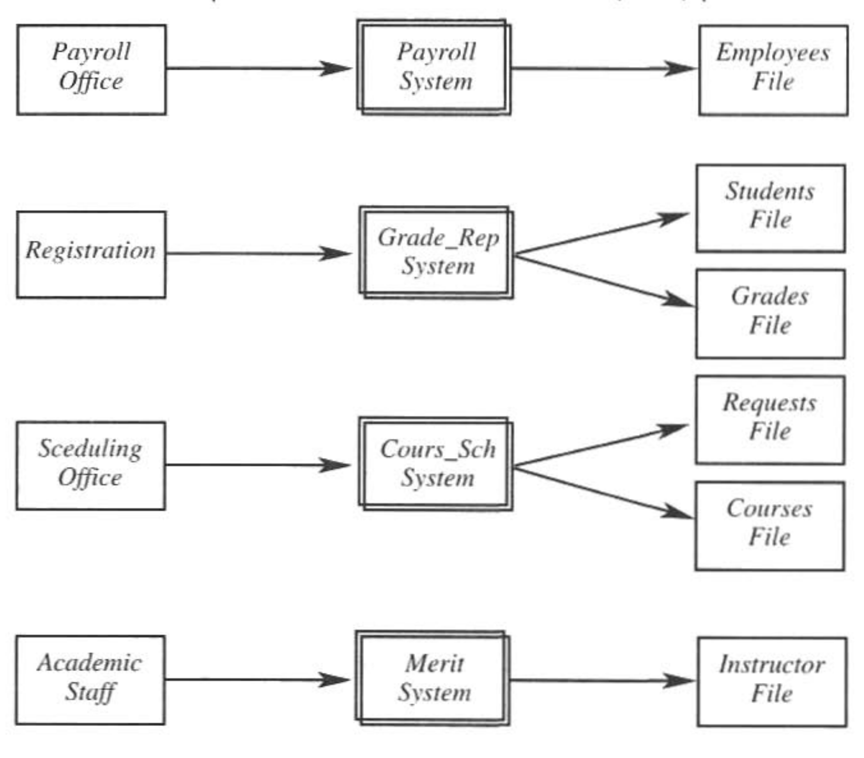
مثال (١–١) :

لنأخذ المثال التالى لقاعدة بيانات خاصة بجامعة افتراضية university database كما هو موضح بالشكل رقم (١–٢). حيث إن شئون الموظفين Payroll office تستخدم ملف "الموظفين" EMPLOYEES لنظام المرتبات. وتستخدم إدارة القبول والتسجيل Registration ملفين، هما ملف "الطلاب" STUDENTS وملف "التقديرات" GRADES لبيان تقديرات الطلاب فى نهاية كل فصل دراسى. وتقوم إدارة البرامج بجدولة المناهج الدراسية Scheduling office فى ملفى "المناهج الدراسية" COURSES و"المتطلبات" REQUESTS. وتستخدم إدارة شئون أعضاء هيئة التدريس Academic Staff ملف "المحاضرين" INSTRUCTORS لحساب مستحقاتهم merit.

لو فرض أن هذه النظم تعمل بشكل منفصل وبطريقة صحيحة ، فإن معرفة التقديرات Grades السابقة أو حساب متوسط التقدير بالنقاط Grade point Average (GPA) لطلاب فصل معين فى وقت جدولة Schedule مواد دراسية معينة ، يكون من الصعوبة بأى حال الحصول على المعلومات المطلوبة ، حيث إن التقديرات توجد فى ملف "الطلاب" STUDENTS لدى إدارة القبول والتسجيل فى حين أن جدولة المواد الدراسية توجد فى إدارة البرامج التى تتعامل مع ملفى "المناهج الدراسية" COURSES

وملف "المتطلبات" REQUESTS ؛ ومن ثم يصعب استخلاص المعلومات رغم توافرها فى الملفات خصوصًا فى حالة البرمجة بلغة مثل الكوبول COBOL أو البسكال Pascal وغيرها. ويزيد الأمر صعوبة أيضًا أن الإجابة على الاستفسار باستخدام تلك اللغات تكون معقدة. أما فى حالة تفادى تلك الصعوبة عنها طريق تكرار البيانات فى الملفات ، فإن ذلك قد يؤدى إلى عدم تناسقها بشكل مؤقت حيث إن كلاً من إدارة شئون الموظفين وإدارة شئون أعضاء هيئة التدريس تتضمن معلومات عن Instructor ؛ ومن ثم فى حالة قيام إدارة شئون الموظفين بتعديل البيانات فسوف تتضارب مع ما هو موجود فى إدارة أعضاء هيئة التدريس ؛ مما يؤدى إلى أخطاء بالبيانات نتيجة تضاربها، وبالتالى تفقد البيانات تناسقها.

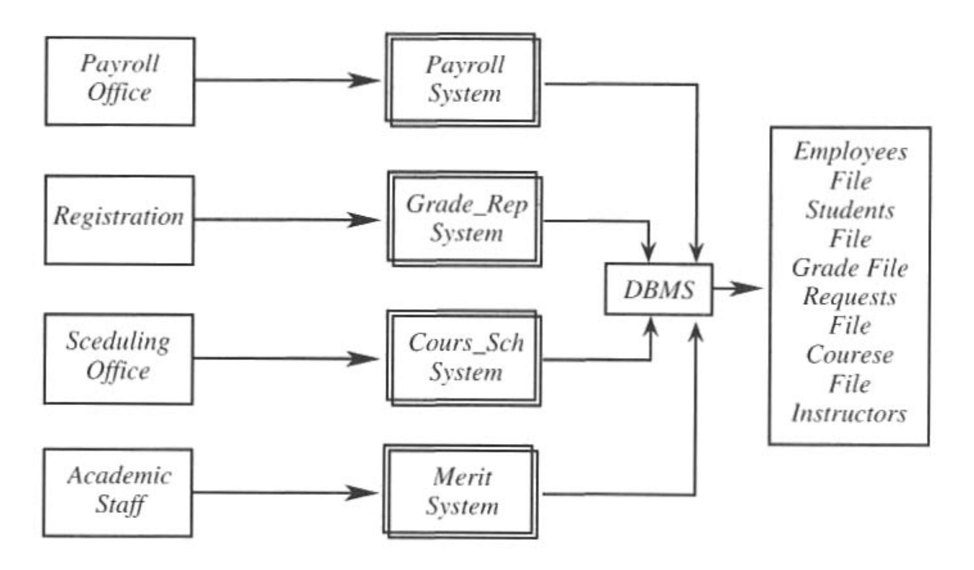
شكل رقم (١ -٢) قاعدة بيانات خاصة بجامعة ذات نظم غير مترابطة



أما فى حالة تناول البيانات داخل هذه الملفات باستخدام نظم إدارة قواعد البيانات DBMSs ؛ فإن البرامج يجب أن تضع واجهة مشتركة interface مع نظم إدارة قواعد البيانات DBMSs لكى يتم الحصول على البيانات ، وذلك لأن نظم إدارة قواعد البيانات DBMSs تتناول كل البيانات بشكل متكامل Integrity وتكامل البيانات يؤدى إلى زيادة الأمن Security للبيانات .

وتوافق البيانات وتناسقها Consistency وتكاملها integrity يصبح سهلاً فى حالة عدم تكرار البيانات(٣) كما هو موضح بالشكل رقم (١-٣) .

شكل رقم (١-٣) قاعدة بيانات خاصة بجامعة باستخدام نظام إدارة قواعد البيانات DBMS

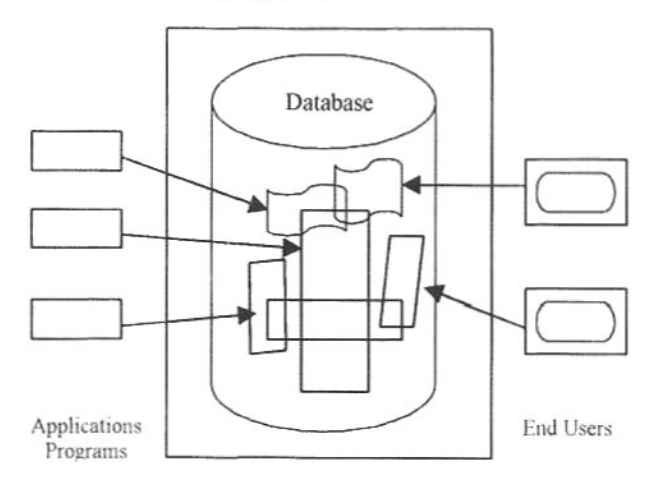


محتويات نظام قاعدة البيانات:

يتضمن نظام قاعدة البيانات بشكل مبسط أربعة محتويات رئيسية، هى : البيانات Data والأجهزة Hardware والبرمجيات Software والمستخدمون Users كما هو موضح بالشكل رقم (١-٤) وسوف يقتصر السرد على كل من البيانات والبرمجيات فقط دون الخوض فى المجالين الآخرين(٤) .

شكل رقم (١-٤) محتويات نظام إدارة قواعد البيانات DBMS





البيانات Data :

يجب أن تكون البيانات متكاملة integrated وموضع مشاركة shared : لذا يلزم توضيح كل من مفهومى التكامل والمشاركة :

التكامل integration :

يتم تكامل البيانات فى قواعد البيانات من خلال توحيد العديد من ملفات البيانات المميزة مع إلغاء البيانات الزائدة عن الحد redundancies بين هذه الملفات بحذف الخصائص attributes المتكررة : لتلك الملفات مما يؤدى إلى تقليل البيانات داخل نطاق الخاصية الواحدة domain .

المشاركة sharing :

ينصب مفهوم المشاركة فى قاعدة البيانات على مشاركة المستخدمين لنفس البيانات لأغراض مختلفة فى نفس الوقت ، وترجع هذه المشاركة فى الحقيقة إلى تكامل البيانات .

قاعدة البيانات Database :

تمثل قاعدة البيانات مجموعة ملفات مترابطة للبيانات الدائمة persistent data التى يتم معالجتها واستعمالها بواسطة نظم التطبيقات application systems وتختلف بيانات قاعدة البيانات من نوع لآخر، حيث توجد بيانات خاصة بالإدخال input data وبيانات خاصة بالمخرجات output data وأوامر التحكم control statements وبيانات الاستعلامات queries وغيرها.

فوائد استخدام قواعد البيانات Benefits of DBs :

١- تقليل تكرار البيانات Redundancy .

٢- تجنب تضارب البيانات Inconsistency .

٣- مشاركة البيانات Shared data .

٤- الالتزام بالمقاييس المتعارف عليها Standards .

٥- ضمان أمن البيانات Security restrictions .

٦- حفظ سلامة البيانات Integrity .

٧- استقلال البيانات Data independence .

نظم إدارة قواعد البيانات DBMSs .

النظام البرمجى هو مجموعة من البرامج المترابطة التى يستخدمها الحاسب لمعالجة البيانات لاستخلاص المعلومات المطلوبة، والبرمجيات Software المعقدة والتى تسمح بالتخزين الملائم للبيانات داخل قاعدة البيانات، وتسمح بإنشاء وحفظ علاقات الربط بين السجلات والملفات فى قاعدة البيانات وتسمح أيضًا للمستخدمين بعمل الاستفسارات للبيانات المخزنة داخل قاعدة البيانات تسمى بنظم قواعد البيانات Database System أو بنظم إدارة قواعد البيانات DBMSs. وهذه البرمجيات المعقدة والتى يتم من خلالها التعامل مع متطلبات المستخدمين التى تصدر استخدام لغة استعلام خاصة (مثل SQL). تعترض نظم إدارة قواعد البيانات DBMSs المتطلبات

وتقوم بتحليلها ثم فحصها أثناء تحولها من المخطط الخارجى للمستخدم المخطط المفاهيمى ثم المخطط الداخلى ، وذلك مروراً بالخرائط التناظرية mapping بين كل مخطط والذى يليه. وأخيراً تقوم نظم إدارة قواعد البيانات DBMSs بتنفيذ العمليات الضرورية على قاعدة البيانات المخزنة.

وظائف نظم إدارة قواعد البيانات DBMS :

تتضمن وظائف نظم إدارة قواعد البيانات DBMSs دعماً للنقاط الآتية(٥):

تعريف البيانات Data Definition :

يجب أن تكون نظم إدارة قواعد البيانات DBMSs قادرة على قبول تعريف البيانات فى شكل المصدر Source وتحويلها إلى شكل المترجم Object . على سبيل المثال : على نظم إدارة قواعد البيانات DBMSs فهم سجل ملف "الموظفين" EMPLOYEES EMP الخارجى الذى يتضمن حقل (خاصية) المرتب Salary ؛ ومن ثم يكون لديها القدرة على التفسير والاستجابة لمتطلبات المستخدم الذى قد يسأل عن المرتب الأقل من ٢٠.٠٠٠$ مثلاً .

معالجة البيانات Data Manipulation :

يجب ان تكون نظم إدارة قواعد البيانات DBMSs قادرة على معالجة متطلبات المستخدم من استرجاع retrieve أو تحديث البيانات update سواء بتعديل modify أوحذف delete للبيانات الموجودة فى قاعدة البيانات أو اضافة insert بيانات جديدة .

أمن البيانات وسلامتها Data Security & Integrity :

يجب أن تتضمن نظم إدارة قواعد البيانات DBMSs قواعد تحول دون انتهاك المستخدمين لقواعد أمن وسلامة البيانات .

استرداد البيانات وتزامنها Data Recovery & Concurrency :

تحكم نظم إدارة قواعد البيانات DBMSs استرداد البيانات وتزامنها عادة عن طريق ما يسمى مدير المعاملات transaction manager .

قاموس البيانات Data Dictionary :

تمدنا نظم إدارة قواعد البيانات DBMSs بقاموس للبيانات عن كل المخططات schemas والخرائط التناظرية mappings والبيانات داخل النظام .

الأداء Performance :

يجب أن تكون نظم إدارة قواعد البيانات DBMSs قادرة على تنفيذ كل الوظائف السابق ذكرها .

البناء المعمارى لنظم إدارة قواعد البيانات ANSI/SPARC Database Architecture :

إن هيكل نظم قواعد البيانات Database Systems الذى اقترح من قبل ANSI/ SPARC والذى تم تجهيزه بواسطة المعهد القومى الأمريكى للمقاييس American National Standard Institute (ANSI) فى أوائل السبعينيات لوضع النظم القياسية لنظم قواعد المعلومات. ومع ذلك ليس كل نظم قواعد البيانات تؤكد ذلك الهيكل . وتوضح الفكرة الرئيسية لهذا الهيكل فى الشكل رقم (١-٥)، وهى خريطة تبين التنقل بين مختلف مستويات هياكل قواعد البيانات الثلاثة. وهذه المستويات هى المستوى الخارجى External والمستوى المفاهيمى Conceptual والمستوى الداخلى Internal[٤)،(٧] .

المستوى الخارجى External View :

عبارة عن هيكل قاعدة البيانات كما يظهر للمستخدم النهائى End-user ويوصف بواسطة المخطط الخارجى External Schema لقاعدة البيانات. ويعتبر ذلك توصيفًا فرعيًا من التوصيف المفاهيمى، ويتم عادة توصيف كل خاصية كما وصفت من قبل فى المستوى المتوسط (المفاهيمى). ومع ذلك يمكن توصيف خصائص attributes غير ممثلة فى المستوى المتوسط ، ولكن يجب أن تعتمد أو تحسب من البيانات الموجودة فى المستوى المتوسط. وقد تختلف الخصائص فى هذا المستوى عن نظائرها فى المستوى المتوسط فى الترتيب.

المستوى المتوسط Conceptual view :

يحوى هذا المستوى كل هيكل قاعدة البيانات المنطقى دون التركيز على الشكل المادى. ويجب أن يحتوى هذا المستوى على كل الجداول tables وثيقة الصلة فى قاعدة البيانات وتمثيل علاقات الربط بينهم، وأحيانًا كثيرة يطلق على هذا المستوى اسم جماعة المستخدمين Community user. يتم توصيف جميع الملفات فى هذا المستوى بحيث يشار إلى نوع كل خاصية . وتكتب القيود Constraints التى تحافظ على سلامة البيانات وتحدد علاقات الربط بين الملفات.

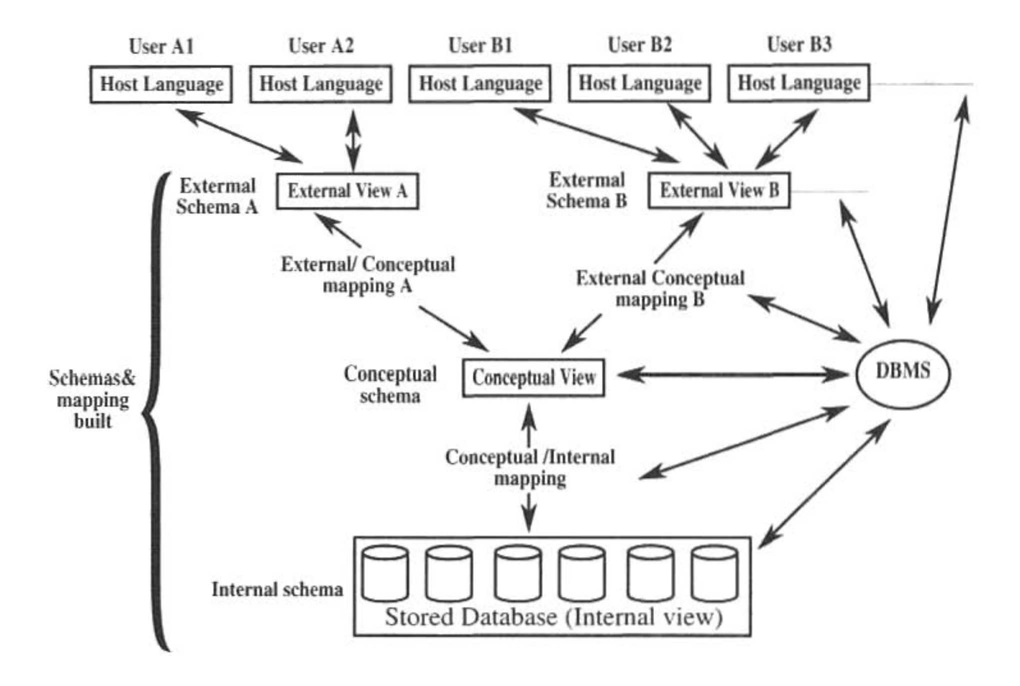
المستوى الداخلى Internal view :

على الرغم من أن المستوى المتوسط يمثل هيكل قاعدة البيانات المنطقى إلا أنه يحصل على المعلومات الخاصة بالملفات من الهيكل الداخلى المادى الذى يشكل هيكل قاعدة البيانات المادية المخزنة. ومن الملاحظ أن الملفات الموصفة بالمستوى الخارجى يتم الحصول عليها من توصيف المستوى المتوسط؛ لذا فإن الاستفسارات أو أوامر التحديث التى تكتب فى المستوى الخارجى ينبغى أن تترجم إلى أوامر المستوى المتوسط والتى بدورها تترجم إلى المستوى الداخلى.

الخرائط التناظرية Mapping :

يوجد توصيفان للخرائط التناظرية: أحدهما من المستوى الخارجى المتوسط والآخر من المستوى الداخلى. ومن خلال توصيف هذه الخرائط يتم تعريف الملفات الموجودة بالمستوى الأعلى إلى توصيف الملفات الموجودة بالمستوى الأقل. ثم يتم مناظرة بين كل من الخصائص فى كلا المستويين. ويجب معرفة أن الفصل بين توصيف المستوى المتوسط عن توصيف المستوى الداخلى، يعنى أن التوصيف المنطقى لقاعدة البيانات لا يتطلب تغيير داخل هياكل التخزين المادى فى المستوى الداخلى، حيث يتم تغيير التوصيف فى المستوى المتوسط؛ ومن ثم يتبعه تغيير تلقائى فى المستوى الداخلى من خلال الخرائط التناظرية. كما أن البرامج المكتوبة لتوصيف المستوى المتوسط لا تتطلب تغييراً. ويؤدى الفصل بين توصيف المستوى الخارجى وتوصيف المستوى المتوسط إلى الاستقلال المنطقى للبيانات.

شكل رقم (٥) يوضح البناء المعمارى لنظام إدارة قواعد البيانات DBMSs



إمكانيات نظم إدارة قواعد البيانات The capabilities of DBMSs :

١- يوجد إمكانيتان للتمييز بين نظم إدارة قواعد البيانات عن غيرها من النظم البرمجية فى القدرة على إدارة البيانات الدائمة :

- تقوم نظم إدارة قواعد البيانات DBMSs بالتعامل مع محتويات قاعدة البيانات وتديرها، وذلك فى حالة وجود قاعدة البيانات بشكل دائم.
- القدرة على التعامل مع أكبر كم من البيانات بكفاءة.

تتميز نظم إدارة قواعد البيانات عن نظم الملفات التى أيضاً تدير بيانات دائمة فى تداول أجزاء سريعة للبيانات بشكل عفوى . وتكمن قدرات نظم إدارة قواعد البيانات DBMS عندما تكون كمية البيانات ضخمة ؛ لأنه فى حالة وجود كميات البيانات القليلة فإن أى تقنية بسيطة مثل الفحص الخطى تكون عادة ملائمة.

٢- ومن إمكانيات نظم إدارة قواعد البيانات DBMSs شائعة الانتشار ما يلى:

- دعم لنموذج بيانات واحد على الأقل أو التجريد الرياضى abstraction mathematical الذى من خلاله يمكن توصيف البيانات.
- دعم لغة من لغات البرمجة ذات المستوى العالى High-level التى تسمح للمستخدم بتعريف هياكل البيانات والتعامل معها ومعالجتها.
- القدرة على إدارة المعاملات transaction Management ، بحيث يتمكن العديد من المستخدمين فى أى لحظة من التعامل مع البيانات بشكل متزامن.
- عدم السماح للمستخدمين الذين لا يمتلكون تفويضاً بالتعامل مع البيانات، وذلك عن طريق التحكم فى تداولها.
- القدرة على عمل مراجعة للتأكد من صحة البيانات.
- القدرة على معالجة أخطاء النظام دون فقدان البيانات.

كفاءة التعامل مع الملف :

توفر نظم إدارة قواعد البيانات DBMSs على الأقل نموذجًا للبيانات يسمح للمستخدم برؤية المعلومات بشكل أكثر فهمًا. ويمكن عادة رؤية البيانات فى مستويات عديدة، حيث تسمح نظم إدارة قواعد البيانات DBMSs برؤية البيانات الأقل نسبياً فى شكل ملفات مدمجة.

مثال (١-٢) :

بفرض أن هناك رغبة فى الاحتفاظ بملف "الموظفين" EMPLOYEES لشركة معينة. بحيث يحتوى هذا الملف على حقلين، هما : اسم الموظف name و اسم المدير manager. ومن ثم يمكن توصيف هيكل السجلات باستخدام لغة شبيهة بباسكال Pascal-Like مثلاً كالتالى:

```
Record
Name  : char (30);
Manager  : char (30);
End ;
```

ومعلوم أن الملف هو مجموعة سجلات متتابعة ؛ ومن ثم فهو يحتوى على سجل لكل موظف بالشركة. وتظهر قدرة نظم إدارة قواعد البيانات DBMSs عند تداولها لبيانات الملف. فعند البحث عن مدير اسمه "أكرم صلاح"، فى حالة كون ملف "الموظفين" EM-PLOYEES يحتوى على آلاف السجلات للموظفين، فإن نظم إدارة قواعد البيانات DBMSs تساعد فى تجهيز "فهرس الملفات". ومن ثم يكون لكل ملف مفهرس معرف وحيد لايتكرر. كما فى حالة فهرسة الملف، حيث يكون لكل سجل معرف مميز له، يسمح بتداول السجل المطلوب مباشرة دون البحث فى كل سجلات الملف. كذلك فى حالة إضافة سجل جديد أو حذف آخر أو تعديل بيانات سجل. وأيضًا تمكن نظم إدارة قواعد البيانات DBMSs المستخدم من التبحر فى الحصول على المعلومات من أكثر من ملف فى حالة وجود علاقات ربط بين هذه الملفات.

لغات الاستعلام Query languages :

تزود نظم إدارة قواعد البيانات DBMSs المستخدمين بلغة استعلام أو لغة معالجة بيانات DML، لتنفيذ العمليات المطلوبة على الملفات. وهنا تجدر الإشارة إلى أن لغات الاستعلام المبنية على أساس نموذج البيانات العلاقى تتطلب تفاصيل أقل من تلك الموجودة فى نماذج البيانات الأخرى.

مثال (١-٣) :

مع افتراض وجود ملفين: أحدهما خاص "بالموظفين" EMPLOYEES والآخر خاص "بالإدارات" DEPARTMENTS. بحيث يحتوى الملف الأول على سجل يتكون من حقلين، هما الاسم Name والإدارة Dept والآخر يحتوى على سجل يتكون من حقلين، هما اسم الإدارة Dept واسم المدير Manager. فإنه يمكن التعبير عن الملفين فى مخطط علاقى كالآتى :

EMPLOYEES (Name , Dept)

DEPARTMENT (Dept , Manager)

عند البحث عن المدير " أكرم صلاح" فإنه ينبغى التبحر من ملف "الموظفين" -EM PLOYEES إلى ملف "الإدارات" DEPARTMENT مستخدمين التساوى بين حقل اسم الإدارة Dept فى كلا الملفين كما هو موضح فى الشكل رقم (١-٦) ؛ وذلك لأن حقل اسم الإدارة Dept يعتبر حقل الربط المشترك بين الملفين.

```
(1) SELECT MANAGER
(2) FROM EMPLOYEES
(3) WHERE EMPLOYEES. NAME = "أكرم صلاح "
(4) AND EMPLOYEES. DEPT = DEPARTMENT. DEPT
```

شكل رقم (١-٦) مثال على استعلام باستخدام SQL

وتبين السطور فى الشكل رقم (١-٦) الخاص بالاستعلام السابق باستخدام لغة الاستعلام البنائية SQL الآتى :

سطر (١) يخبر نظام إدارة قواعد البيانات DBMS أن يختار خاصية اسم المدير Manager.

سطر (٢) يخبرنظام إدارة قواعد البيانات DBMS أن ينظر إلى ملف "الموظفين" EMPLOYEES.

سطر (٣) يخبرنظام إدارة قواعد البيانات DBMS أن اسم الموظف هو "أكرم صلاح".

سطر (٤) يخبر نظام إدارة قواعد البيانات DBMS أن المدير يرتبط بالموظف بكونه موجوداً فى نفس الإدارة فى جدول "الإدارة" DEPARTMENT مع نفس الإدارة فى جدول "الموظفين" EMPLOYYEES. فى الشكل رقم (١-٧) نفس الاستعلام للنموذج الشبكى باستخدام لغة معالجة البيانات DML :

```
(1) EMPLOYEES. NAME : "أكرم صلاح "
(2) FIND EMPLOYEES RECORD BY CALC.KEY
```

(3) FIND OWNER OF CURRENT EMP.DEPT SET

(4) FIND FIRST MANAGER RECORD IN CURRENT DEPT.MGR SET.

(5) PRINT MANAGER. NAME

شكل رقم (١-٧) مثال على استعلام باستخدام لغة معالجة البيانات DML

سطر (١) ، (٢) يخبر نظام إدارة قواعد البيانات DBMS أن البحث عن السجل الخاص بالموظف " أكرم صلاح" فى ملف "الموظفين" EMPLOYEES.

سطر (٣) يستعمل تركيبة فئة set الخاصة بموظف - إدارة EMP-DEPT ، والتى تمثل علاقة الربط التى تربط الموظفين بإدارتهم لإيجاد الإدارة التى ينتمى إليه الموظف.

ملاحظة : owns , set مصطلحات فنية تستخدم داخل لغة معالجة البيانات DML فى النظام الشبكى.

سطر (٤) يوجد فئة set أخرى خاصة بالإدارة - مدير DEPT-MGR ، والتى تمثل علاقة الربط التى تربط الإدارات بالمديرين.

سطر (٥) يطبع أول مدير اسمه "أكرم صلاح" فى قائمة الأسماء.

يلاحظ هنا التبحر الواضح فى الاستعلام باستخدام لغة معالجة البيانات DML فى النظام الشبكى عنها فى لغة الاستعلام البنائية SQL ، ولكن الصعوبة الكبرى تواجه مخطط البرامج فى كتابة لغة معالجة البيانات DML فى النظام الشبكى عنها فى لغة الاستعلام البنائية SQL. وعند المقارنة بين الشكلين السابقين ليس من جهة عدد السطور الزائدة فى لغة معالجة البيانات DML فى النظام الشبكى ؛ بل من جهة الفرق الكامن فى عدم التمكن من الحصول إلا على سجل واحد فقط به الاستفسار فى حالة استخدام لغة معالجة البيانات DML فى النظام الشبكى ، أما فى حالة استخدام لغة الاستعلام البنائية SQL فإنه يتم الحصول على بيانات مترابطة لأكثر من سجل . وهذا أحد الأسباب المهمة التى تؤدى إلى استخدام لغة الاستعلام البنائية SQL المبنية على أساس النموذج العلاقى.

لغات قاعدة البيانات Database Languages :

توجد الأوامر Statements الخاصة بالتوصيف declarations فى لغات البرمجة التقليدية والأوامر القابلة للتنفيذ فى كل جزء للغة البرمجة. أما فى نظم قواعد البيانات فيتم الفصل بين وظيفتى التوصيف والمعالجة فى لغتين مختلفتين. توجد البيانات فى لغة البرمجة التقليدية فقط حين يتم تنفيذ البرنامج فى حين أن البيانات فى نظم قواعد البيانات توجد بشكل دائم ، وربما يتم توصيفها مرة واحدة فقط. وفيما يلى توضيح مبسط للغات قواعد البيانات[٦] .

لغات تعريف البيانات DDL :

يوصف المخطط المفاهيمى Conceptual schema بلغة تسمى لغة تعريف البيانات Data Definition Language (DDL) ، ويتم تعريفها كجزء من نظام إدارة قواعد البيانات DBMS. وتستخدم لغة تعريف البيانات DDL لتوصيف أنواع الكينونات (الملفات) Entity Types ، وعلاقات الربط relationships بينها، وهى ليست لغة إجرائية Procedural Language.

مثال (١-٤) :

توضح الأوامر التالية إنشاء الجدول العلاقى (ملف) relation الذى يصف ملف "الرحلات الجوية" FLIGHTS بواسطة لغة تعريف البيانات DDL:

```
CREATE TABLE FLIGHTS ( NUMBER : INT, DATE : CHAR(8),
SEAT : INT, FROM : CHAR(3), TO  : CHAR (3)) ;
CREATE INDEX  FOR  FLIGHTS ON NUMBER ;
```

ويمثل المثال السابق تشفيراً Coding خاصاً بلغة تعريف البيانات DDL للغة الاستعلام البنائية SQL. يصف الأمر الأول الجدول العلاقى relation والخصائص attributes الخاصة به فى حين يصف الأمر الثانى الفهرسة على رقم الرحلة flight ، مع ملاحظة أنه يفصل بين كل أمر وآخر الفصلة المنقوطة (;). ويتم استعمال لغات تعريف البيانات DDL عند تصميم قاعدة بيانات أو عند تعديل تصميم سابق وتشمل مجموعة

أوامر للتوصيف. أما التصميم المفصل لقاعدة البيانات المادية يتم بواسطة برامج نظام إدارة قواعد البيانات DBMS التى تترجم الأوامر فى لغات تعريف البيانات.

لغة معالجة البيانات DML :

تتطلب العمليات الخاصة بقواعد البيانات لغة مخصصة لمعالجة البيانات Data Manipulation Language (DML) أو لغة استعلام للتعبير عن الأوامر مثل :

١- استرجاع البيانات من قاعدة البيانات Retrieve .

٢- تحديث للبيانات بقاعدة البيانات Updates .

وينفذ التحديث بواسطة أوامر خاصة فى لغة الاستعلام البنائية SQL. ومثال ذلك :

```
UPDATE FLIGHTS
SET SEAT = SEAT - 4
WHERE NUMBER = 123 AND DATE = "AUG 31 ";
```

٣- إضافة بيانات لقاعدة البيانات بمعنى إضافة سجل داخل قاعدة البيانات ويعبر عنه بواسطة برنامج فى SQL مثل :

```
INSERT INTO FLIGHTS
VALUES (356, "AUG18 ", 100, ORD ", JFK );
```

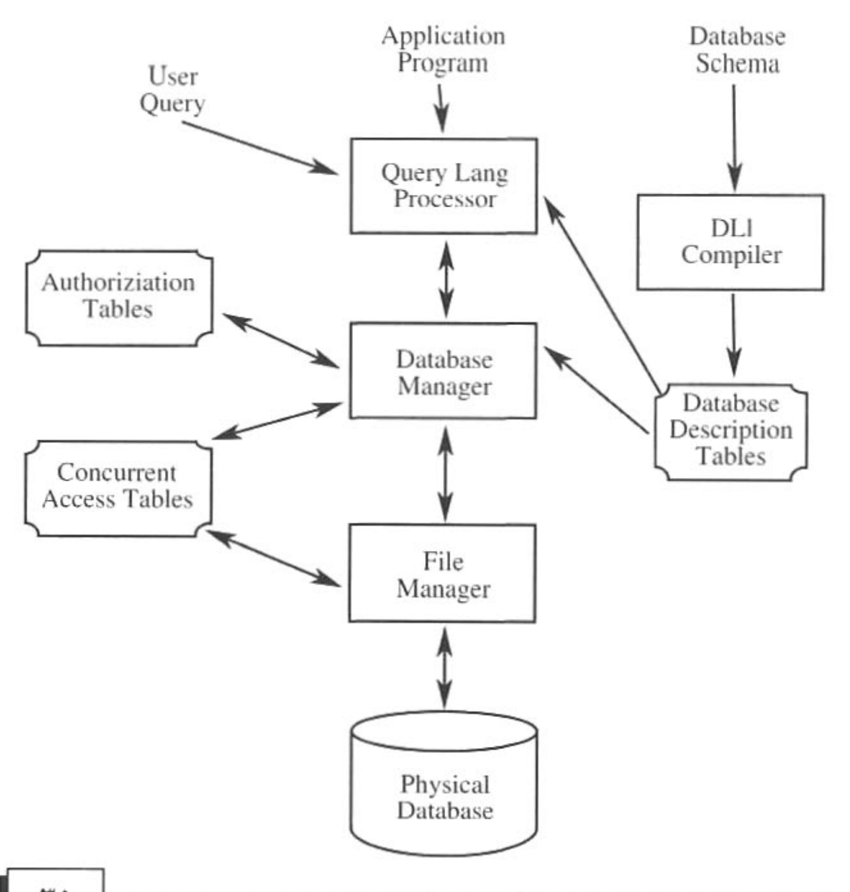
لغة المضيف التقليدية Host Language :

تكتب البرامج الخاصة بمعالجة قواعد البيانات بلغة تقليدية مثل لغة سى C ولغة الكوبول COBOL وغيرها، ويطلق على مثل هذه اللغة لغة المضيف. وتستخدم لغة المضيف لعمل كل شيء مثل اتخاذ القرار وعرض الاستفسارات. وتضمن أوامر لغة معالجة البيانات DML فى برامج لغة المضيف بإحدى الطريقتين التاليتين معتمدة على طبيعة نظام إدارة قواعد البيانات DBMS مثل :

١- أوامر تكتب فى لغة معالجة البيانات DML بواسطة لغة المضيف فى صورة إجراءات Procedures ملحقة بنظام إدارة قواعد البيانات DBMS.

٢- أوامر تمثل امتداداً للغة المضيف ، بحيث يوجد مترجم وسيط Preprocessor لتبادل أوامر معالجة البيانات أو مترجم Compiler لكل من لغة المضيف وأوامر لغة معالجة البيانات DML. تحول أوامر لغة معالجة البيانات DML إلى نداءات Calls للإجراءات بواسطة نظام إدارة قواعد البيانات DBMS.

شكل رقم (١-٨) رسم تخطيطى لنظام قاعدة البيانات



محتويات نظم قواعد البيانات :

يبين الشكل رقم (١-٨) تفاعل مختلف محتويات نظم قواعد البيانات، حيث تبين الجهة اليمنى التصميم أو مخطط قاعدة البيانات التى تغذى مترجم لغة تعريف البيانات DDL الذى يقدم توصيفًا ماديًا لقاعدة البيانات، و نادرًا ما يتم تعديل مخطط قاعدة البيانات مقارنة مع كل من الاستفسارات ولغة معالجة البيانات DML التى تنفذها. كما يوضح الشكل السابق معالج لغة الاستعلام الذى يحصل على برامج معالجة البيانات من مصدرين. أحدهما: استعلامات المستخدم أو معالجات البيانات الأخرى التى يتم إدخالها مباشرة من الوحدة الطرفية. ثانيهما: البرامج التطبيقية ، حيث تكون استعلامات قواعد البيانات والمعالجات مضمنة فى لغة المضيف host language التى يتم معالجتها مسبقًا؛ لكى تنفذ جزءًا من هذه البرامج التطبيقية أخيرًا والتى تكون مكتوبة بلغة تقليدية يتم تناولها بواسطة مترجم لغة تقليدية. ويكون جزء من البرنامج التطبيقى أوامر لغة معالجة البيانات التى يتم تناولها بواسطة معالج لغة الاستعلام والذى يكون مسئولاً عن جعل هذه الأوامر أقرب ما تكون إلى الفاعلية. وينبغى التأكيد على أن أوامر لغة معالجة البيانات DML تستخلص البيانات من قاعدة البيانات التى يتم تحويلها معنويًا بواسطة معالج الاستعلام Query processor. يصل معالج الاستعلام إلى توصيف جداول قاعدة البيانات التى تنشىء بواسطة برنامج لغة تعريف البيانات DDL لتتحقق من بعض الحقائق التى تكون مفيدة لجعل الاستعلامات أقرب ما يكون إلى الفاعلية، مثل وجود أو عدم وجود فهرس Index. ويأخذ مدير قاعدة البيانات Database Manager الأوامر من المستوى المفاهيمى ويحولها إلى أوامر فى المستوى المادى أى إلى مستوى الملفات. ويحتفظ مدير قاعدة البيانات بالجداول ويعمل على الوصول إلى جداول معلومات التفويض وجداول معلومات ضبط التزامن، حيث تسمح جداول معلومات التفويض لمدير قاعدة البيانات أن يراجع صلاحية المستخدم الخاص بتنفيذ الاستعلام المستهدف أو تعديل قاعدة البيانات. كما أن تعديل جدول معلومات التفويض يتم بواسطة مدير قاعدة البيانات. لو تم التداول المتزامن لقاعدة البيانات بواسطة الاستعلامات أو معالجات قواعد البيانات فإن مدير قاعدة البيانات يحتفظ بالمعلومات الضرورية فى جدول خاص.

ويترجم مدير قاعدة البيانات الأوامر التى يحصل عليها إلى عمليات Operations مرتبطة بالملفات التى يتم تداولها بواسطة مدير الملفات. ويستعمل مدير الملفات جداول التداول المتزامن؛ لكى يسمح لأكثر من عملية أن تتداول قاعدة البيانات بشكل متزامن[٧] .

نظم أساس الشىء Object-base system :

يستخدم مصطلح أساس الشىء Object-base ومصطلح نظم إدارة قواعد البيانات الشيئية الموجهة Object-Oriented Database Management system (OO-DBMS) لوصف نوع من نظم البرمجة بإمكانيات نظم إدارة قواعد البيانات ، وفى كل من لغتى معالجة البيانات والمضيف تتوافر المعالم الآتية :

١- الأشياء المعقدة Complex Objects :

وتعنى القدرة على تعريف أنواع بيانات مع هيكل متداخل كنموذج البيانات الذى تبنى فيه أنواع البيانات على شكل سجلات أو مجموعات، والذى ينشئ هياكل متداخلة ويعتبر أكثر الأساليب انتشاراً. مثال ذلك تبنى القيم المرتبة (الصف) tuple من أنواع أساسية مثل الأرقام الصحيحة والحقيقية وسلاسل الحروف وغيرها لتشكل هيكل السجل. والجدول العلاقى Relation يبنى من قيم مرتبة (صفوف) tuples لتشكل مجموعة، بمعنى أن الجدول العلاقى هو مجموعة قيم مرتبة (صفوف) set of tuples لتشكل سجلات خاصة. ونستطيع إنشاء سجل واحد ذى محتويات من نوع مجموعة قيم مرتبة (صفوف)، وهى تمثل هياكل أكثر تعقيداً.

٢- الكبسلة Encapsulation :

وتعنى القدرة على تعريف إجراءات تلحق فقط بنوع معين من الأشياء Objects ، والقدرة على التعامل مع هذه الأشياء عن طريق تطبيق أحد هذه الإجراءات. ومثال ذلك عندما نعرف الركام Stack بوصفه نوعًا من أنواع البيانات، ويمكن تعريف عمليات معينة تلحق بهذا النوع مثل الدفع Push، والسحب Pop، وكذلك عند تعريف الملف File بوصفه نوعاً من أنواع البيانات ، يمكن أن تلحق به عمليات القراءة Read والكتابة Write.

٣- هوية الشيء Object Identity :

ويقصد بها قدرة النظام على تمييز شيئين لهما نفس المحتويات من النوع الأساسى . والأنواع الأساسية كما سبق ذكرنا مثل الأرقام بأنواعها ، والحروف وسلاسل الحروف وغيرها .

والنظام الذى يدعم هوية الشىء يشار إليه بوصفه شيئًا موجهًا Object-Oriented بالرغم من أن مصطلح الشيء الموجه يتضمن دعمًا لتجريد نوع البيانات Abstract Data Type (ADT). أما النظام الذى لا يدعم هوية الشيء سوف يطلق عليه قيمة موجهة Value-Oriented أو سجل موجه؛ ولذا كانت كل النظم المبنية على النماذج العلاقية للبيانات ذات قيمة موجهة.

فى حين أن النظم التبحريه لقواعد البيانات المبنية على النماذج الشبكية والهرمية هى نظم شيئية موجهة فى معنى محدود جداً لدعم هوية الشىء عند التمييز بين هيكل الملف file ومخطط العلاقة EMPLOYEES (Name, DEPT)؛ حيث إن تعريف الملف يتوافق مع هوية الشىء؛ لأن موضع السجل يميزه عن أى سجل آخر. بينما عند النظر إلى البيانات كجدول علاقى relation ، توجد صعوبة فى تخزين البيانات فى قيم مرتبة مرتين(صفين) Two tuples كل منهما يحتوى على نفس القيمة المرتبة. ويرجع السبب فى ذلك إلى أن الجدول العلاقى هو فئة Set. والقواعد التى تحكم نظرية الفئات فى علم الجبر العلاقى تمنع تكرار القيمة المرتبة (الصف) Tuple. أما عن النظام الذى يدعم الكبسلة والأشياء المعقدة يقال عنه أنه يدعم تجريد أنواع البيانات (ADT) أو أنواع الأشياء classes. بمعنى أن تجريد أنواع البيانات هو تعريف لبنية معينة مع تعريف عمليات على الأشياء ؛ لذلك يمكن أن تعالج نوع الشىء class.

الهوامش :

1- [GILLENSON, 1990], Mark L. Gillenson, 'Physical Design Equivalencies in Database Conversion',**Communications of the ACM**, Vol.33, Number 8. August 1990.

2- [LIEBERHERR, 1993], Karl Lieberherr, and Cun Xiao, 'Formal Foundations for Object-Oriented Data Modeling', **IEEE Transactions on Knowledge and Data Engineering**, Vol. 5, No.3, June 1993.

3- [GRANT, 1987], John Grant, **Logical Introduction to Databases**, Harcourt Brace Jovanovich, Publishers and its subsidiary, Academic Press, 1987.

4- [DATE,1995], C.J. Date, An Introduction to Database Systems 'Volume I, sixth edition, Addison-Wesley publishing Company Inc.,1995.

5- [DATE,1990], C.J. Date, **An Introduction to Database Systems**, Volume I, fifth Eedition, Addison-Wesley publishing Company Inc.,1990.

6- [ULLMAN,1988], Ullman J.D., **Principles of Database and Knowledge-base System**, Vol. 1, Computer Science Press, 1988.

7- [Connolly 1996], Thomas M. Connolly and Carolyn E. Begg, **Database Systems**, Addison-Wesley Publishing Co., Inc., 1996.

الفصل الثانى

تصنيف نماذج البيانات

مقدمة :

سوف يتم مناقشة عدد من الموضوعات المتعلقة بتصنيف نماذج البيانات فى هذا الفصل. وفيما يلى استعراض موجز للنقاط الرئيسية التى سيتم التطرق إليها :

نموذج البيانات :

يشير نموذج البيانات إلى النظام المستخدم لنمذجة البيانات بطريقة معينة. ويتم استخدامه عادة لتوصيف هيكل البيانات.

مجال نماذج البيانات :

يتم استخدام نماذج البيانات بصفة دائمة لنمذجة الصفات الساكنة للبيانات. وتتضمن هذه الصفات هيكل البيانات والقيود الخاصة بها والتى تتضمن قيدين أساسيين هما :

- قيود السلامة التى تحكم الحالات الصحيحة لقاعدة البيانات.
- قيود أمن وحماية البيانات من سوء الاستعمال.

كما سيتم التعرف على معالم نماذج البيانات التى توضح بعض التفاصيل الإضافية التوضيحية لاستعمال البيانات كالتوزيع والأمن والبيانات الزائدة عن الحد ومنظورات نماذج البيانات.

خطوات تصميم قاعدة البيانات :

تتكون هذه الخطوات من ثلاثة نماذج ، هى :

- نموذج التصميم المفاهيمى لمخطط قاعدة البيانات الذى يوفر القدرة على فهم البيانات.
- نموذج التصميم المنطقى الذى يعمل على تنظيم البيانات فى نموذج قابل للتطبيق.
- نموذج التصميم المادى الذى يتم استخدامه لتوصيف تخزين البيانات.

تصنيف نماذج البيانات :

سوف يتم تصنيف نماذج البيانات الخاصة بقواعد البيانات فى اتجاهين أساسيين: أحدهما يرتبط بنشاط تصميم قاعدة البيانات والآخر خاص بتصنيف أجيال نماذج البيانات.

تطبيق نماذج البيانات :

يتم وضع محتويات نماذج البيانات فى ملفات عند التنفيذ من خلال تطبيق النماذج التقليدية والتى تشمل النماذج الهرمية والشبكية والعلاقية، وكذلك من خلال تطبيق نموذج البيانات الشيئية الموجهة الذى يتطلب التحول من تمثيل الأشكال البنائية للنظام نحو توصيف سلوك متكامل لكل من هيكل البيانات والعمليات الخاصة به.

الصعوبات التى تواجه إدارة قواعد البيانات :

هناك العديد من الصعوبات التى تواجه إدارة قواعد البيانات ، مثل تكرار البيانات والسجلات متغيرة الطول ومفاتيح البيانات الثانوية.

تطبيق علاقات الربط فى نظم قواعد البيانات :

هناك ثلاثة أنواع لعلاقات ربط البيانات هى:

- واحد – لواحد.
- واحد – لمتعدد.
- متعدد – لمتعدد.

ولكن من الصعوبة بأى حال تطبيق علاقة الربط متعدد – لمتعدد عند تطبيق علاقات الربط فى نظم قواعد البيانات؛ لذا فإنه من الضرورى تحويل هذه العلاقة إلى واحد – لمتعدد؛ ومن ثم سيتم التطرق للسياسات المتبعة فى علاقة الربط واحد – لمتعدد سواء سياسة الملفات متعددة الوصلات أو سياسة الملفات المعكوسة.

تمثيل علاقات الربط فى قواعد البيانات التقليدية :

سوف يتم توضيح كيفية تمثيل علاقات الربط فى قواعد البيانات التقليدية فى كل من النماذج التبحرية سواء الهرمية أو الشبكية ، والنماذج العلاقية.

نموذج البيانات Data Model :

هو مجموعة متكاملة من المفاهيم لوصف بيانات مؤسسة أو هيئة معينة وتوضيح العلاقات بينها والقيود عليها والغرض منه هو تمثيل تلك البيانات و جعلها مفهومة. ويستخدم عادة لتوصيف هيكل قاعدة البيانات وتزويد قاعدة البيانات بالعمليات Operations الخاصة بها. ويقصد بهيكل قاعدة البيانات أنوا ع البيانات Data types وعلاقات الربط relationships بينها والدلاليات Semantics عليها والقيود Constraints التى تضمن سلامة البيانات ودقتها والتى تعرف بقوالب templetes تلك القاعدة. أما عن العمليات التى ينبغى على النموذج أن يوفرها لقاعدة البيانات فتشتمل على الاسترجا ع retrieval والتحديث updating الذى يتضمن الإضافة insertion والحذفdeletian والتعديل modification وغيرها. وللربط مع المفاهيم السابقة يجب التفرقة بين مصطلحين ، هما: مصطلح نموذج البيانات والآخر هو مصطلح نموذج التطبيق. حيث يشير مصطلح نموذج البيانات إلى النظام المستخدم لنمذجة البيانات بطريقة خاصة ، ويوفر كتل البناء building blocks أو بنيات النمذجة modeling constructs التى يمكن بها وصف هيكل قاعدة البيانات. فى حين يشير مصطلح نموذج التطبيق Application model إلى توصيف البيانات لقاعدة بيانات خاصة. على سبيل المثال النموذج العلاقى هو نموذج بيانات ، وتعريف قاعدة بيانات خاصة لتطبيق شخصى فى شركة معينة هو نموذج تطبيق لقاعدة البيانات تلك التى تستخدم النموذج العلاقى. ومن المهم فى أى تطبيق أنه التمييز بين توصيف قاعدة البيانات وقاعدة البيانات فى حد ذاتها ، حيث إن التوصيف يشير إلى مخطط قاعدة البيانات Database Schema ، ويصمم مخطط قاعدة البيانات لمجموعة من التطبيقات من خلال تحليل المتطلبات. ويتم توصيفها باستعمال نموذج بيانات محدد من خلاله يوفر بنيات النمذجة فى شكل تركيبة لغوية معينة language syntax أو شكل تخطيطى مناسب. أما فى البيئات التى تتميز بفاعلية مستمرة مثل التصميم باستخدام الحاسب (CAD) ، التصنيع باستخدام الحاسب (CAM) ، أدوات

هندسة البرمجيات (CASE) ، التحكم فى الوقت الحقيقى أو نظم المعلومات الجغرافية (GIS) فإن مخطط المنتج المصمم قد يغير نفسه ؛ ومن ثم يشار إليه بوصفه مخططاً تطورياً Schema evolution.

مجال نماذج البيانات Scope of Data Models :

تستخدم نماذج البيانات لتصميم مخطط قاعدة البيانات التى تكون محدودة فى مجال استخدامها. كما تستخدم هذه النماذج لنمذجة الصفات الساكنة للبيانات ، والتى تتضمن هيكل البيانات والقيود عليها ومعالمها ومنظوراتها views (١) :

١- هيكل البيانات Structure of data :

يعبر هيكل نموذج البيانات عن كيفيه تجميع أنواع البيانات ذات القيمة الواحدة atomic data types فى أنواع الترتيب الأعلى .order – higher ، وأكثر من ذلك تعبر النماذج عن العلاقات بين هذه التجمعات ومن هذه التجمعات aggregates :

– السجل الموجه record - oriented :

يشكل هيكل التجميع البنية الأساسية لنوع السجل فى كونه يتكون من مجموعة حقول . ومخطط قاعدة البيانات المبنية على السجل الموجه تتكون من عدد من أنواع السجلات المترابطة بطرق مختلفة .

– نموذج البيانات الهرمى hierarchical data model :

تميز النموذج الهرمى بتنظيم أنواع السجلات الموجهة فى هيكل شجرى (هرمى) .

– نموذج البيانات الشبكى network data model :

تميز هذا النموذج بتنظيم أنواع السجلات الموجهة كنقاط التقاء على رسم شبكى تربط بين كل نقطة وأخرى وصلة edge تمثل علاقة الربط بين السجل والآخر .

– النموذج العلاقى Relational data model :

قدم هذا النموذج رؤية للفئة – الموجهة Set - riented لنمذجة البيانات كفرع من فروع الجبر العلاقى relational algebra .

– النموذج الشيئى الموجه Object-oriented data model .

هيكل هذا النموذج قاعدة البيانات فى حدود الأشياء Objects والأشياء المتداخلة in-terobjects فى التفاعلات والتى تتم بشكل شائع.

٢– القيود الخاصة بنماذج البيانات Constraints on data models :

تمثل القيود حصراً إضافيًا على أنواع السجلات أو الوقائع Instances داخل قاعدة البيانات. وقيود نموذج البيانات تخدم هدفين :

– قيود السلامة Integrity :

وتُعَدُّ قيود السلامة القواعد التى تحكم الحالات الصحيحة لقاعدة البيانات. وهى إما أن تكون أنواعًا أساسية للبيانات مثل الأرقام الصحيحة أو سلاسل الحروف أو الحروف وغيرها، أو تكون قواعد تم تعريفها بواسطة المستفيد لكى تعكس معنى البيانات.

– قيود الأمن والحماية Security and protection :

وتطبق هذه القيود على قاعدة البيانات لحمايتها من سوء الاستعمال ومن الاستعمال بدون تصريح ويمكن رؤية القيود فى مستويات مختلفة :

(أ) القيود الوراثية Inherent Constraints :

تخص القيود التى تبنى داخل قواعد نموذج البيانات نفسه. مثل نموذج كينونة– علاقة Entity-Relationship (ER) ، بحيث يجب أن تكون العلاقة قائمة بين نوعى كينونتين على الأقل Two entity types. وسوف يرد شرح تفصيلى لهذا النموذج فى الفصل الخامس.

(ب) القيود الضمنية Implicit constraints :

يتم توصيف هذه القيود لنموذج البيانات باستخدام لغة تعريف البيانات DDL ؛ لكى تصف صفات أخرى إضافية. مثال ذلك قيد المساهمة الإجبارى فى نموذج كينونة – علاقة (ER) الذى يبين أن نوع كينونة معينة ينبغى أن تسهم فى علاقة خاصة.

(ج) القيود الصريحة Explicit Constraints :

هى قيود تبعية التطبيق التى تصف القيود الدلالية التى ترتبط مع هذا التطبيق. وهى قيود عامة وصعبة فى وصفها بتفاصيل كاملة. ويوجد اتجاه عام يؤكد على اعتبارها "سلوك تطبيق" للمعلومات داخل قاعدة البيانات؛ لكى يتم الإستيلاء عليها فى موضع معين. تهدف لغة 4 GL إلى الاستيلاء على دلاليات التطبيق فى المستوى الأعلى، ونظم إدارة قواعد البيانات حاليًا لا تتناولها بسهولة. وهناك بعد آخر لنمذجة القيد؛ لكى يتم الاستيلاء على حالة التحول أفضل من حالة المعلومات الساكنة . وهذا يعطى دفعًا للقيود التى تتميز بفاعلية مستمرة dynamic Constraint ، والتى تكون صحيحة فى حدود أى أنواع للتغييرات على قواعد البيانات. مثالاً على ذلك : زيادة مرتب "موظف معين" من الممكن أن يزيد كلاً من القيود الساكنة والقيود ذات الفاعلية المستمرة التى تسمح بتعريف قاعدة بيانات متناسقة Consistent. فى حين تقيم القيود الساكنة على "نقطة" سريعة لقاعدة البيانات؛ لكى تحدد صحتها، فإن القيود ذات الفاعلية المستمرة تكون أكثر صعوبة؛ لأنها تشتمل على عوائق لحالة التحول فى "وقت التنفيذ run-time" .

٣- معالم نماذج البيانات features of data Models :

يصف نموذج البيانات الخاص بقاعدة بيانات أيضًا بعض التفاصيل الإضافية التوضيحية لاستعمال البيانات :

(أ) التوزيع Distribution :

أحد المعالم التى ترتبط بجزء من البيانات فى حدود إمكانية تخزين البيانات كجزيئات، ومثال على ذلك فى النموذج العلاقى الذى أصبح معتاداً أن يشار إلى جزئياته أفقيًا ورأسيًا . Horizantal and vertical fragments ففى الجدول العلاقى الخاص "بالموظفين" EMPLOYEES فإن كل جزئية أفقية قد تحتوى على بيانات الموظف الخاصة به، فى حين أن الجزيئات الرأسية لنفس الجدول العلاقى الخاص "بالموظفين" EMPLOYEES قد تشتمل على معلومات عن أسماء الموظفين والإدارات الخاصة بهم .

(ب) الأمن Security :

يُعَدُّ أحد المعالم المهمة التى قد تبنى فى نموذج البيانات فى مختلف المستويات.

(ج) البيانات الزائدة عن الحد Redundancy :

تُعَدُّ أحد المعالم التى يصعب أن يتم نمذجتها بشكل واضح. وذلك قد يسيطر عليها فى النموذج الخاص بالنسخ الصريحة أو تداخل البيانات.

(د) المفاتيح Keys :

أحد هذه المعالم المهمة التى يجب أن يسمح النموذج فيها بتوصيف مثل هذه المفاتيح التى لا تتكرر Unique Keys فى تعريف البيانات، والتى يكون لها القدرة على توصيف المعاملات المادية Physical Parameters مثل التجميع clustering أو الفهرسة indexing مثل شجرة الفهرس B+ على حقل معين كجزء من توصيف نموذج التطبيق.

٤- منظورات نماذج البيانات Views of data models :

يقصد بالمنظورات فى نمذجة قواعد البيانات أنها نموذج تطبيقى يتم تعريفة بواسطة مستخدم معين أو مجموعة مستخدمة لتطبيق ما. وتوفر معظم نماذج البيانات لغة لتعريف المنظورات Views ربما من خلال لغة الاستعلام، حيث يتم الحصول على الناتج من الاستعلام معرفًا كمنظور. ويحبذ إنشاء المنظور فى حالة وجود طلب استرجاع معلومات منه أفضل من تجسيد المنظور؛ لأن تجسيد المنظور ينشئ بيانات زائدة عن الحد.

خطوات تصميم قاعدة البيانات Steps of Database Design :

عملية تصميم قاعدة البيانات تتكون من رسم تفصيلى لمتطلبات البيانات والتطبيقات بشكل ناجح خلال الخطوات الموضحة بالشكل رقم (٢-١) الخاص بعملية تصميم مخطط قاعدة البيانات [١][٢] .

١- التصميم المفاهيمى Conceptual Design :

يوفر لنا هذا النموذج القدرة على فهم الإدراك الحسى للمستخدمين تجاه البيانات . ولهذا النموذج القدرة على توفير المفاهيم الضرورية لدعم بيئة التطبيق فى أعلى مستوى بلا توصيف للنظام nonsystem-specific.

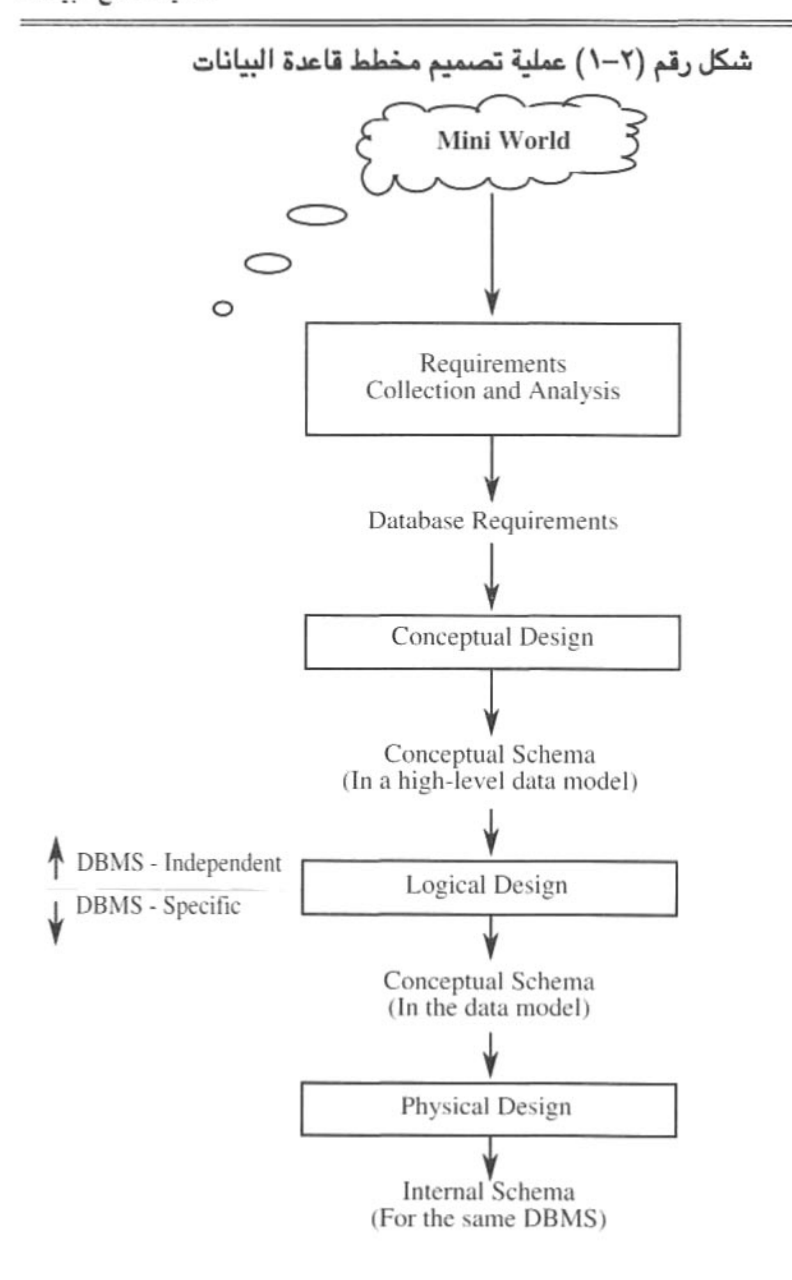
٢- التصميم المنطقى Logical Design :

يعمل هذا النموذج على تنظيمً البيانات فى نموذج البيانات القابل للتطبيق ويعرف باسم نموذج البيانات المنطقى أو بالنموذج التنفيذى Implementation Model. ويمتلك هذا النموذج هياكل نمذجة فى حالة ما إذا تم اتباعها من قبل المستخدم ؛ لتمكن من تجنب الخوض فى التفاصيل المادية (العملية) للتطبيق ، ولكن قد يحبذ المستخدم الحصول على نتيجة تنفيذ الحاسب مباشرة فى بعض نظم قواعد البيانات.

٣- التصميم المادى Physical Design :

يحبذ فى هذه المرحلة ألا تستعمل نمذجة البيانات فى توصيف البيانات. حيث يتكون التصميم المادى من اختيارات مختلفة لتخزين البيانات فى حدود التجميع Clus-tering والتقسيم Partitioning والفهرسة Indexing وغيرها.

شكل رقم (٢-١) عملية تصميم مخطط قاعدة البيانات



تصنيف نماذج البيانات Classification of data models :

يمكن تصنيف نماذج البيانات الخاصة بقواعد البيانات بشكل أساسى فى اتجاهين رئيسين [٣]،[٤] :

أولاً : الاتجاه الأول يتعامل مع الخطوات الخاصة بنشاط تصميم قاعدة البيانات والتى تطبق النموذج فى الخطوات السالف ذكرها ، وهى تشمل :

- التصميم المفاهيمى .
- التصميم المنطقى .
- التصميم المادى .

ثانيًا : الاتجاه الآخر ويمكن أن يصنف إلى أربعة أجيال :

١- نماذج البيانات الأولية Primitive Data Models :

يتم تمثيل الأشياء Objects فى هذه النماذج بهياكل السجل المجمعة فى هيكل الملف. والعمليات الرئيسية المتوافرة هى القراءة Read والكتابة Write والعمليات الأخرى الخاصة بالسجلات .

٢- نماذج البيانات التقليدية Classical Data Models :

تشمل هذه النماذج من جانب : النماذج التبحرية Novigational Models والتى تشمل كلاً من النماذج الهرمية والنماذج الشبكية. ومن جانب آخر تتضمن النماذج العلاقية.

(أ) النماذج التبحرية Navigational Models :

تتضمن النماذج التبحرية كلاً من النموذج الهرمى والشبكى . ويعتبر النموذج الهرمى امتداداً لنموذج البيانات الأولى (البدائى). ويمثل النموذج الشبكى أيضًا امتداداً للنموذج الهرمى. ويعبر عن هيكل البيانات فى هاتين النموذجين فى حدود البيانات التى يمكن تجزيئها؛ بمعنى البيانات ذات القيمة الواحدة فقط atomic data.

وتجمع أنواع هذه البيانات ذات القيمة الواحدة فى الترتيب الأعلى higher-order ، وتتجه لتكون السجل والذى يتكون بدوره من حقول. وهنا مخطط قاعدة البيانات يتكون من عدد من السجلات التى ترتبط معًا بطرق مختلفة. يتم تنظيم السجلات فى النموذج الهرمى فى شكل شجرى tree structure فى حين أن تنظيم السجلات فى النموذج الشبكى فى شكل رسم شبكى يتضمن رؤوسًا vertices تمثل السجلات ووصلات edges تمثل علاقات الربط بين هذه السجلات.

(ب) النموذج العلاقى Relational Model :

اعتمد هذا النموذج على الأساس الرياضى فى نمذجة البيانات المبنية على أساس نظرية الفئات Sets theory والجداول العلاقية relations فى علم الجبر العلاقى. وتتكون الجداول العلاقية من قيم مرتبة (صفوف) tuples يتم تعريفها على مجموعة خصائص attributes. يجب أن تكون هذه الخصائص معرفة مسبقًا على نطاق معين من القيم Domain of values. وبسبب سهولة نمذجتها قد حصلت على أنتشار واسع بين مطورى التطبيقات[٧]. يتضمن هذا النموذج مجموعة من العمليات Operations نتيجة ارتباطه بعلم الجبر العلاقى منها: عمليات الاختيار الافقى SELECT والاختيار الرأسى PROJECT والربط JOIN وكذلك بقية العمليات فى نظرية الفئات الخاصة بالاتحاد UNION والتقاطع INTERSECTION والفرق DIFFERENCE والضرب الكارتيزى CARTESIAN PRODUCT وهكذا. جعلت هذه العمليات النموذج العلاقى، أكثر قوة لأن كل الجداول تصبح معاملات argnments للعوامل العلاقية وقد بنيت بعض لغات الاستعلام مثل SQL، QUEL على أساس الحساب العلاقىRelational Calculus والذى أصبح معيارًا لصناعة معالجة البيانات.

ويوجد نوعان من القيود على هذا النموذج : أولهما يسمى قيد سلامة الكينونة entity integrity وثانيهما قيد السلامة المرجعية referential integrity ، واللتان سوف يتم توضيحهما فيما بعد. والعديد من نظم إدارة قواعد البيانات العلاقية جيدة التصميم أدت إلى توفير تداول النموذج العلاقى، منها: INGERS . RDb, DB2, SYBASE . IN- FORMIX, ORACLE .

٣- نماذج البيانات الدلالية Semantic Data Models (SDMs) :

تحتفظ النماذج التقليدية بارتباطها بالسجل، ومعنى المعلومات فى قواعد البيانات التقليدية ليس واضحًا. ولهذا السبب تحاول نماذج البيانات الدلالية جعل المعنى أكثر تعبيرًا فى تمثيل معنى المعلومات عما هو متوافر فى النماذج التقليدية.

كما أنه يستحيل مع أساسيات النموذج العلاقى أن يتم الحصول والتحكم فى كثير من الدلاليات للتطبيق المعقد من خلال إطاره البسيط. مثالاً على ذلك قدرة النموذج العلاقى على تأكيد قيد السلامة المرجعية ولكن لا توجد آلية للتميز بين مختلف أنوا ع علاقات الربط التى قد توجد بين أنوا ع الكينونات، مثل علاقات الربط متعدد - لمتعدد ، وتصميم علاقات الربط واحد - لمتعدد ، وصفات علاقات التبعية - الموجودة فى واحد - لمتعدد. وإيجاد القدرة على صنع مثل هذا التمييز يمكن من تعريف دلاليات للعمليات الخاصة بإنشاء وحذف علاقات الربط بشكل مختلف لكل حالة.

ويمثل نموذج كينونة - علاقة Entity-Relationship (ER) نموذجًا للبيانات الدلالية وقد تم تطويرة مرات عديدة وتم تدعيمه بمفاهيم عن نوع الشىء class ونوع الشىء الفرعى subclass والوراثة inheritance الهرمية المبنية على أساس التعميم gener-alization والتخصيص specialization. وكانت الفكرة الأساسية من وراء هذا الدعم هى أن يشتمل نموذج كينونة - علاقة ER على كل تجريدات البيانات Abstraction of Data وسُمِّى بنموذج كينونة - علاقة المطور Enhanced Entity-Relationship (EER) . وتجدر هنا الإشارة إلى طرح سؤال : ماذا يحدث فى حالة تجاهل الفرق بين نوع الكينونة وعلاقة الربط ؟. بتأمل بسيط يمكن الإجابة على هذا فى أنه يتم الحصول على نموذج عام يحتوى على أنوا ع الأشياء classes (Object types) .

٤- نماذج البيانات المرتبطة بالشىء Object-Oriented Data Models :

هناك العديد من أوجه التشابه والاختلاف بين مصطلحات كل من نماذج البيانات الدلالية ونماذج البيانات الشيئية الموجهة والتى سوف يرد شرحها بإسهاب فى الفصول القادمة . والتى يمكن ذكرها فيما يلى بشكل مختصر :

(أ) أوجه التشابه :

- بنية التجريدات والقيم .
- القيم وأنواع الدمج .
- التعريف الداخلى والخارجى للشىء .

(ب) أوجه الاختلاف :

- مدى الوراثة .
- إخفاء المعلومات وتغليفها (كبسلتها) encapsulation .

تطبيق نماذج البيانات Implementation of Data Models :

تعرض نماذج البيانات كميات كبيرة من البيانات فى تنظيم مبسط ولكن توضع محتويات هذه النماذج فى ملفات عند التنفيذ من خلال نظم قواعد البيانات[٥].

(١) تطبيق النموذج الهرمى :

تم تطوير النموذج الهرمى للبيانات كجزء من نظم إدارة قواعد البيانات بواسطة شركة IBM تحت اسم نظام إدارة المعلومات Information Management System (IMS) . ولا توجد أى وثائق أو تقارير من أى مجموعة من المجموعات التى قامت بتعريف نموذج البيانات الهرمى ، ولكن استخدام نظام إدارة المعلومات IMS للنموذج الهرمى للبيانات جعله مرتبطًا بنموذج نظام إدارة المعلومات IMS للبيانات . ولقد أصبح نظام إدارة المعلومات IMS نظامًا مهمًا جدًا ؛ لأن كثيرًا من المنظمات قد هيئت أجهزتها به. ويسمح نموذج البيانات الهرمى لمخطط قاعدة البيانات أن يتم هيكلتها على شكل شجرة تمثل رؤوسها nodes أنواع السجلات record types والوصلات links بينها هى علاقات الربط relationship بين الأب والابن Parent-child بين هذه السجلات . ومع ذلك فنظام إدارة المعلومات IMS يمكن تحويله إلى نموذج شبكى بشكل محدود وذلك بجعل نوع السجل له سجلان أبويان Two porents أحدهما يسمى الأب المادى (الحقيقى) Plysical Parent والآخر يسمى الأب المنطقى (الوهمى) Logical Parent.

(٢) تطبيق النموذج الشبكى :

تم تحليل نموذج البيانات الشبكى بالتفاصيل من قبل اقتراح "لمجموعة مهام قاعدة البيانات" DataBase Task Group (DBTG) ، والتى قدمته تحت اسم النموذج التشاورى للغة نظام البيانات COnference on DAta SYstem Language (CODASYL). ويتم تمثيل هيكل البيانات فى النموذج الشبكى برسم شبكى فيه الرؤوس nodes التى تمثل أنواع السجلات record types، والوصلات edges تمثل علاقات الربط relationships بين هذه السجلات، ويتم تمثيلها بعلاقات الربط من نوع واحد - لمتعدد one-to-many ويطلق عليها أنواع الفئات set-types .

والنظم التى تنفذ هذه النماذج التبحرية (الهرمية والشبكية) لديها لغة تعريف البيانات التى تعرف قاعدة البيانات ، ولغة أخرى لمعالجة البيانات فى الاسترجاع والتحديث . عدد كبير من نظم إدارة قواعد البيانات DBMSs قد تم تنفيذها باستعمال هذه النماذج ولازالت مستخدمة حتى الآن ولا سيما فى البنوك وشركات الطيران والمكتبات ومعظم الشركات الضخمة التى كان لديها إمكانيات مالية عالية وبدأت فى استخدام الحاسب فى تعاملاتها مبكرًا. ويرجع استمرار هذه الشركات فى استخدام هذه النماذج إلى دقة وصحة النتائج والتقارير المستخرجة بواسطتها وتأديتها للأغراض التى صممت من أجلها على أكمل وجه.

(٣) تطبيق النموذج العلاقى :

ارتبط تعريف نموذج قاعدة البيانات العلاقى بالمؤلف أ.ف. كودد .E. F. Codd. الذى وضع مفاهيم قواعد البيانات العلاقية، حيث بين الخلفية النظرية للنموذج العلاقى فى نظرية الفئات المرتبطة بالعلاقات الرياضية. وقد استنتج لغات معالجة البيانات الرئيسية التى يتم استعمالها مع النموذج العلاقى والتى جعلت تلك اللغات العمليات المطبقة على قاعدة البيانات أكثر قوة ، وهى :

- الجبر العلاقى relational algebra .
- الحساب العلاقى relational Calculus .

(٤) تطبيق نموذج البيانات الشيئية الموجهة :

يتطلب تطبيق هذا النموذج التحول من تمثيل الأشكال البنائية Structural aspects للنظام نحو توصيف سلوك متكامل Integrated behavioral لكل من الهيكل Structure والعمليات Operations : حيث إن كبسلة هيكل البيانات مع العمليات الخاصة بها يعرف عادة بنوع البيانات التجريدية Abstract Data Type (ADT). ويساعد ذلك على التحكم فى التطبيقات المعقدة والتى غالبًا تزيد بمحاولة تعريف حلول فى مستويات تجريديه غير مناسبة . والعديد من نماذج البيانات الشيئية الموجهة قد نفذت فى نظم مشابهة مثل : O2, IRIS, Gemstone .

الصعوبات التى تواجه إدارة قواعــد البيانــات :

هناك العديد من الصعوبات التى تواجه إدارة قواعـد البيانــات فى تمثيل البيانات. بصفة عامة، قد ينظر للملف كمجموعة قوائم متوازية. على سبيل المثال الشكل رقم (٢-٢) يصور ملف سجلات اللجان COMMITTEES لشركة افتراضية . الخطوط الرأسية تقسم الملف إلى قوائم منفصلة أو حقول، فى حين أن الخطوط الأفقية تقسم الملف إلى قيم منفصلة للسجلات كما سبق وتم إيضاحها[٥] .

شكل رقم (٢ – ٢) يوضح سجلات ملف اللجان لشركة افتراضية

Emp Name	Address	Salary Rate	Tax Category	Ed. Level	Spouse Name
أحمد	Jeddah	9.34	A	12	سلوى
ماجد	Dammam	8.45	B	16	نرمين
محمد	Ryiadh	7.09	A	14	سهير
محمد	Khobar	6.45	C	12	ماجدة
أشرف	Dammam	9.12	A	16	هناء

أ- مشكلة تكرار البيانـات :

تعرف قاعدة البيانات بوصفها مجموعة ملفات مترابطة . ولكن كمثال بسيط لقاعدة بيانات "اللجان" ومزاياها ، بحيث يمكن عرضها فى مقارنة لمعالجة ملف طبيعى لتوضيح مشكلة تكرار البيانات التالية. بفرض أن الشركة الافتراضية تقرر أن لجنة المرتبات ينبغى ألا تحصل على تداول "للمستوى التعليمى" Educational- level للموظف أو "لاسم القرين" (الزوج ـ الزوجة) Spouse. بنفس الطريقة ينبغى ألا تحصل لجنة التخطيط على تداول "معدل المرتب" Salary rate للموظف أو "مصنف الضريبة" Tax category.

طريقة "عدم استخدام قاعدة بيانات" لحل هذه المشكلة تؤدى إلى إنشاء ملفين منفصلين (غير مرتبطين) : أحدهما يحتوى على معلومات مطلوبة من قبل لجنة المرتبات salary committee ، والآخر يحتوى على البيانات المطلوبة من قبل لجنة التخطيط planning committee. وهذا الأسلوب يؤدى إلى تخزين حقل "اسم الموظف" Employee name وحقل "العنوان" Address فى ملفين منفصلين. ولكن ماذا عن دقة تحديث المعلومات التى تم تخزينها بهذا النمط المتكرر؟ على سبيل المثال لو أن موظفًا تم نقله أو تغيير عنوانه فإنه سيضطر إلى أن يوضع فى موضعين مختلفين ويحتمل أن يتم بواسطة شخصين مستقلين. بوضوح فى هذه الطريقة قد ينتج موظفًا يعيش فى عنوانين مختلفين، أحدهما للجنة المرتبات وآخر للجنة التخطيط.

شكل رقم (٢ – ٣) يوضح ملفات قاعدة بيانات اللجان

SALARY FILE		MAIN FILE		PLANNING FILE	
Salary Rate	Tax Cat.	Emp. Name	Address	Ed. Level	Spouse Name
9.34	A	أحمد	Jeddah	12	سلوى
8.45	B	ماجد	Dammam	16	نرمين
7.09	A	محمد	Ryiadh	14	سهير
6.45	C	محمد	Khobar	12	ماجدة
9.12	A	أشرف	Dammam	16	هناء

طريقة "استخدام قاعدة البيانات" لحل مشكلة تكرار البيانات سوف تؤدى إلى تقسيم الملف المبين فى الشكل رقم (٢-٢) إلى ثلاثة ملفات مترابطة كما هو موضح بالشكل (٢-٣) حيث إن الملفات الثلاثة تحل محل ملف اللجان. أحد هذه الملفات ، يحتوى على حقل "اسم الموظف" وحقل "العنوان" ويسمح هذا الملف بالتداول الشائع ويسمى ملف "البيانات الأساسى" MAIN. أما الملف الثانى فهو يحتوى على حقل "معدل المرتب" وحقل "مصنف الضريبة"، ويسمح بالتداول من قبل لجنة المرتبات فقط ويسمى ملف "المرتبات" SALARY . الملف الثالث يحتوى على الحقول المطلوبة للجنة التخطيط ولا يسمح لغير لجنة التخطيط بتداولها ، وتشمل حقلى "مستوى التعليم" و"اسم القرين" ويسمى ملف "التخطيط" PLANNING.

علاقة الربط بين سجلات هذه الملفات (قاعدة البيانات) تكون مباشرة إلى أبعد حد ويمكن تمثيلها بواحد - لواحد ، وهى أبسط نوع من أنوا ع علاقات الربط. بحيث تكون السجلات فى نفس الموضع نسبياً فى مختلف الملفات وتكون مترابطة كل مع الآخر. عندئذ يمكن لعضو لجنة التخطيط أن يستعمل برنامجاً فرعياً للبحث عن اسم موظف فى ملف "البيانات الأساسية" ، ومن ثم تداول نفس المكان فى الملف يكون محدود على استعمال لجنة التخطيط التى تحدد المستوى التعليمى للموظف واسم القرين. يلاحظ أن تحديث المعلومات سوف يتم مرة واحدة وسوف يكون معروف بشكل فورى لكل المستفيدين المخولين لاستعمال تلك المعلومات.

ب- مشكلة السجلات متغيرة الطول :

لو فرض أن هناك نظاماً مسئولاً عن سجلات الطلاب بالجامعة، وأن اسم كل طالب يجب أن يرتبط بعدد متغير للمناهج الدراسية التى يختارها الطالب. وعند تخزين كل البيانات فى ملف واحد، سوف تظهر مشكلة السجلات متغيرة الطول Variable length [٦] كما هو موضح بالشكل رقم (٢-٤) .

شكل رقم (٢ - ٤) يوضح ملف الطالب ذا السجلات متغيرة الطول

STUDENT	COURS 1	COURS 2	COURS 3	COURS 4	COURS 5	COURS 6
خالد نيازى	BUS 44 A	MAT 44 B	PHI 33 A	CPS 11 B		
جيهان فؤاد	MAT11C	ENG 22 D				
هانى هلال	PHI77B	PSY 33 A	COM 99 A	CPS 33 B	ENG 22 B	MAT 33 A

تمثل الوصلة المفرضة فى الجانب الأيمن للبيانات فى الشكل رقم (٢-٤) بوضوح بعض مشاكل تخزين البيانات. والسؤال هنا: كم حجم التخزين المطلوب تخصيصه لسجل كل طالب للتأكد من أن السجل يحتوى على المناهج الدراسية التى اختارها ؟ فى نفس الوقت كيف يمكن تقليل الحيز المفقود إلى أدنى حد ؟ يتضارب السؤلان المطروحان كل مع الآخر بوضوح ، ولا سيما أن الطالب المهنى قد يتطلب مائة حقل للمناهج الدراسية ولكن مثل هذا التخصيص سوف يؤدى إلى فاقد ضخم فى الحيز لمعظم الطلاب لو كان السجل ثابت الطول.

ومن ناحية أخرى فإن الرسم المنظورى لقاعدة البيانات، سوف يسمح برؤية هذه البيانات كملفين منفصلين . أحدهما ملف "الطالب" STUDENT ، والآخر هو ملف "المناهج الدراسية" COURSES. وعلاقة الربط بين ملف "الطالب" وملف "المناهج الدراسية" سوف تكون واحد – لمتعدد، وذلك عكس نوع علاقة الربط واحد-لواحد التى توجد فى المثال السابق لقاعدة بيانات اللجان. وهذا يعنى أن سجل كل طالب يرتبط بعديد من سجلات المناهج الدراسية فى حين أن كل موظف فى ملف "البيانات الأساسى" يرتبط فقط بسجل واحد فى ملف المرتبات وسجل واحد فقط فى ملف "لجنة التخطيط". علاقة الربط المعقدة واحد – لمتعدد سوف تتطلب مؤشرات ، لكى يتم الحفاظ على استمرار علاقة الربط بين ملفات "الطالب" و"المناهج الدراسية" ، كما هو مشار إليه فى الشكل رقم (٢-٥). ويبين هذا الشكل سجلات المناهج الدراسية التى تم اختيارها بواسطة كل طالب كقائمة متصلة Linked List بمؤشر رأسى head Pointer لأنه مخزن فى ملف "الطالب" .

علاقة الربط واحد-لمتعدد الممثلة بين ملفات "الطالب" و"المناهج الدراسية" فى نظام قاعدة بيانات الطالب فى الشكل رقم (٢ – ٥) يمكن أن يأخذ فى الاعتبار تقريباً كل مشاكل إدارة قاعدة البيانات. وحقيقة الأمر أن علاقات الربط الأكثر تعقيداً يتم تنفيذها عادة بتفكيكها إلى علاقات ربط عديدة كل منها واحد – لمتعدد.

شكل رقم (٢-٥) قاعدة بيانات الطالب

ST. FILE

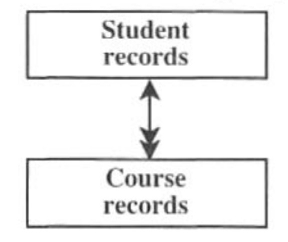
ST.	NAME	LINK
1	خالد نيازي	1
2	جيهان فؤاد	2
3	هانى هلال	3

COURSE FILE

COURSE NAME	GRADE	LINK
BUS44	A	4
MAT11	C	5
PHI77	B	6
MAT44	B	8
ENG22	D	Null
PHY33	A	7
COM99	A	9
PHI33	A	10
CPS33	B	11
CPS11	B	Null
ENG22	B	12
MAT33	A	Null

يوضح الشكل رقم (٢-٦) علاقة الربط فى قاعدة بيانات الطالب ، حيث يبين السهم المزدوج الرأس الذى يشير إلى ملف المناهج الدراسية وجود أكثر من سجل للمنهج الدراسى لكل سجل طالب، فى حين أن السهم الوحيد الرأس الذى يشير تجاه ملف الطالب يشير إلى وجود سجل طالب واحد فقط مرتبط بكل سجل منهج دراسى. يلاحظ أن علاقة الربط واحد - لمتعدد هى فى الحقيقة مثال خاص للشجرة العامة، والتى يمكن رؤيتها بمقارنة المعلومات المخزنة فى قاعدة البيانات فى الشكل رقم (٢-٥) مع الشجرة العامة فى الشكل رقم (٢-٧).

شكل رقم (٢-٦) يبين علاقة الربط واحد - لمتعدد



ج – معالجة المفتاح الثانوى :

يظهر المثال الموضح بالشكل رقم (٢–٨) ملف "الطالب" الذى يوضح طول السجل الثابت لكل طالب وكيفية ظهور علاقة الربط واحد–لمتعدد فى معالجة قاعدة البيانات[٦]. ويتكون سجل الطالب من الحقول التالية :

الرقم المعرف للطالب Identifier (ID) ، اسم الطالب Name ، نوع الطالب Sex ، الصف الدراسى Class. ويستخدم الرقم المعرف للطالب ID كمفتاح لعمل البحث السريع فى هذا الملف ومع ذلك ماذا يحدث لو كانت متطلبات المستفيد هو الاستفسار عن المتطلبات التالية :

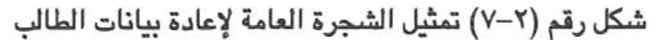
١– إيجاد كل الطلاب من نوع "إناث" F.

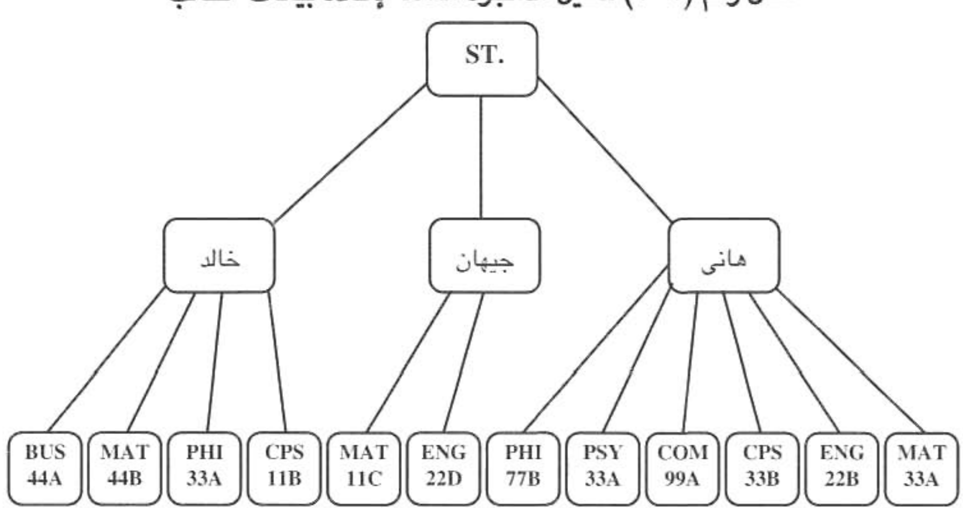
٢– إيجاد كل الطلاب الذين يدرسون بالصف الثانى Sophomore (SO).

٣– إيجاد كل الطلاب من نوع "ذكور" M ويدرسون بالصف الثانى Sophomore (SO).

٤– إيجاد كل الطلاب من نوع "إناث " F أو يدرسون بالصف الثالث Junior (JU).

هذه الاستفسارات تتطلب محاولات لتداول السجلات فى الملف بواسطة مفتاح ثانوى غير وحيد (أى قد يتكرر) . وهذا يعنى أن كل استفسار سابق يتطلب استعمال حقل (أو حقول) بخلاف حقل المفتاح الأساسى "الرقم المعرف للطالب" ID. وهذا الحقل (أو الحقول) يُستعمل مفتاحًا ثانويًا يمكن بواسطته تداول السجلات. أكثر من ذلك، قد يكون التداول المطلوب عبر حقول المفتاح الثانوى له عديد من السجلات تشارك نفس القيمة؛ لذلك يطلق على حقول المفتاح الثانوى اصطلاح غير وحيدة Non-Unique . طبيعة حقول المفتاح غير الوحيدة تعنى أن كل مفتاح ثانوى فى تأثيره يعرف علاقة ربط واحد – لمتعدد بين قيمة المفتاح والسجلات بالملف. على سبيل المثال علاقة الربط واحد – لمتعدد التى توجد فى الملف فى الشكل رقم (٢–٨) والموضحة بالشكل رقم (٢–٩) .


شكل رقم (٢-٧) تمثيل الشجرة العامة لإعادة بيانات الطالب



شكل رقم (٢-٨) يوضح تنظيم سجلات الطلاب حسب حقل المعرف Identifier (ID)

	IDENTIFIER	NAME	SEX	CLASS
1	34762	أحمد على	M	JU
2	37938	إبراهيم طه	M	SE
3	12387	سادة محمد	F	FR
4	27127	علا محمد	F	SO
5	93791	حسن السيد	M	JU
6	35261	آمال فهمى	F	SO
7	59795	منى كمال	F	SO
8	23719	رامى عوض	M	FR
9	64272	أحمد عادل	M	SE
10	48262	محمد السعيد	M	FR
11	58799	شيماء كامل	F	JU
12	97271	سامح السعيد	M	SO
13	59143	رانيا محمد	F	FR
14	87927	دينا مدحت	F	SE
15	28098	شريف محمد	M	JU
16	47819	نانسى نجيب	F	SE

تطبيـــق علاقات الربط فى نظم قواعد البيانـــات :

أ- علاقة الربط واحد - لمتعدد :

السياسة الوحيدة التى يمكن أن تتناول متطلبات المفتاح الثانوى هى التحرك ببطء تتابعًا خلال تنظيم الملف فى الشكل رقم (٢-٨) ، ومراجعة كل سجل لرؤية ما إذا كان يقابل الشرط المحدد. مع ذلك، فى قواعد البيانات ذات العدد الضخم للسجلات، فان طريقة التحرك ببطء تتابعيًا سوف تكون بطيئة جداً من الجهة العملية. ومن ثم فإن المدخل لإدارة قاعدة بيانات فعالة هو [٦] :

١- بدء التخطيط ويعنى التنبؤ بأنواع متطلبات المفتاح الثانوى التى يحتمل استخدامها لقاعدة البيانات التى نحن بصددها الآن.

٢- بناء علاقات الربط واحد - لمتعدد التى يتم تمثيلها باستخدام المفاتيح الثانوية فى هياكل الملفات التى تصنع قاعدة البيانات.

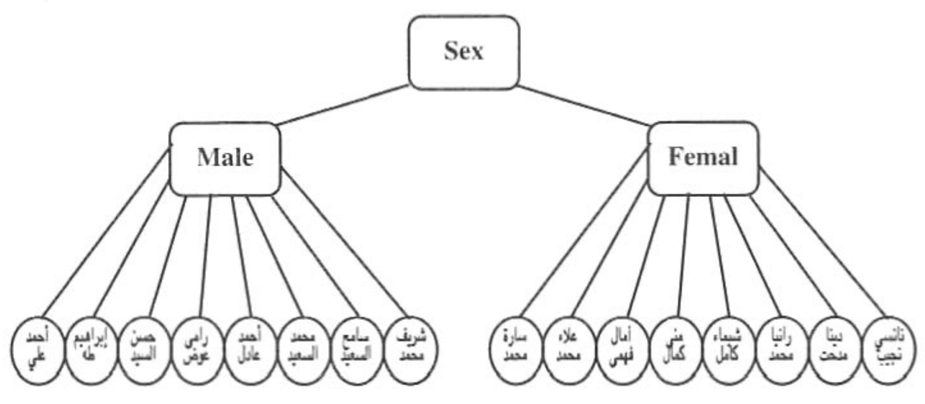
أما عن كيفية بناء علاقات ربط واحد - لمتعدد فى الملفات المكونة لقاعدة بيانات ، فإن الطرق الأكثر شيوعًا تتبع إحدى السياستين التاليتين :

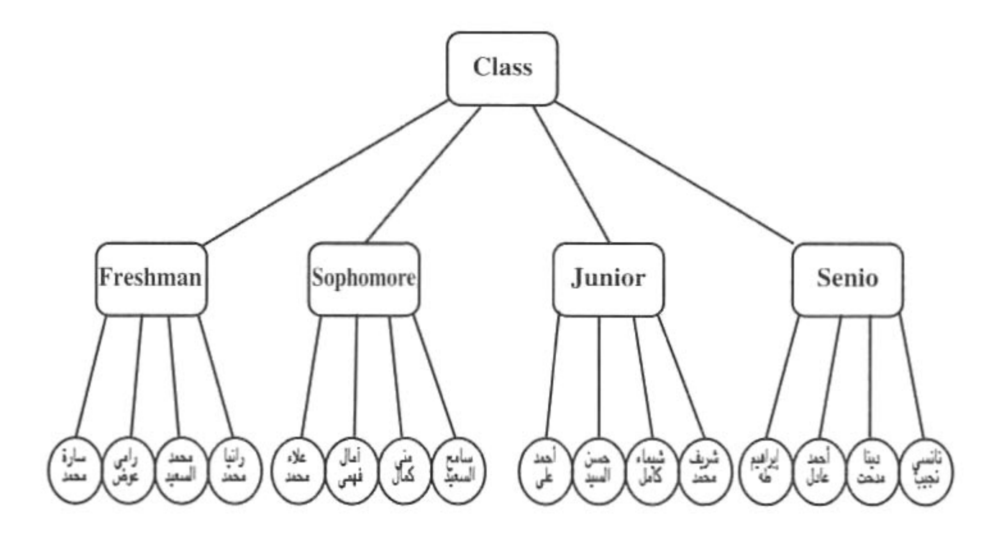
- الملفات متعددة الوصلات . Multilink Files
- الملفات المعكوسة . Inverted Files

أولاً - الملفات متعددة الوصلات :

سياسة الملف المتعدد الوصلات تتطلب حقلاً خاصًا بمفتاح ثانوى يتم بواسطته التعجيل فى تداول الملف، وصلة الحقل يجب أن يتم إنشاؤها فى هيكل السجل للملف . ومن ثم وصلة الحقل تستعمل لتوصيل السجلات التى تشارك فى القيمة المحددة معًا للمفتاح الثانوى فى حالة الاستفسار. وهذا يعنى أن كل مفتاح ثانوى يؤدى إلى العديد من القوائم المتصلة، ولكن قائمة واحدة لكل قيمة قد يلتزم بها المفتاح. وهذه القوائم تنسج أسلوبها خلال قاعدة البيانات، وتسمح للمستفيدين بتداول السجلات التى تعنيهم فقط بكفاءة. ويتم تطبيق تعدد الوصلات لقاعدة بيانات الطلاب فى الشكل رقم (٢-٨) ، الشكل رقم (٢-٩) فى الشكل رقم (٢-١٠).

شكل رقم (٢-٩) علاقة الربط واحد - لمتعدد فى السجلات الموضحة بشكل (٢-٨)





يوجد فى قواعد البيانات، سياسة تطبيق تعدد الوصلات تسمى قاعدة البيانات ذات المؤشر المتسلسل Chained Pointer . بعض المزايا والعيوب النسبية للملف المتعدد الوصلات التى تأخذ فى الحسبان المعرفة المسبقة لهياكل بياناته الضمنية، القائمة المتصلة، والتى يمكن طرحها فيما يلى:

مزايا الملف متعدد الوصلات :

(١) سهولة القدرة على المعالجة التتابعية الجيدة إلى حد بعيد لهذه السجلات التى تحقق قيمة المفتاح الثانوى الوحيد؛ وذلك بتتبع تعرج القائمة المتصلة المناظرة. وأن عدد التداولات المطلوبة للملف هو عدد السجلات التى تحقق قيمة المفتاح الخاص.

(٢) سهولة القدرة على الاحتفاظ بترتيب معين للقوائم المتعددة الوصلات ؛ مما يسهل الإضافة والحذف لتلك القوائم. وهو أحد الاعتبارات التى ينبغى أن تأخذ فى الحسبان إذا ما تم الاحتفاظ بعلاقات ربط واحد - لمتعدد كقوائم متصلة مزدوجة ، والدافع وراء ذلك هو أن الحذف سوف يتطلب عادة تداول السجل لكى يتم حذفه بواسطة المفتاح الأساسى؛ ومن ثم حذفه من كل علاقات الربط واحد - لمتعدد المشارك فيها.

(٣) التغييرات فى قيمة المفتاح الثانوى لسجل معين هى فقط موضوع الحذف من قائمة واحدة متبوعة بإضافة فى قائمة أخرى. على سبيل المثال: النزول من الصف الدراسى الثالث إلى الصف الدراسى الثانى للطالب "حسن السيد" فى الشكل رقم (٢-١٠) خلال محاولة معينة (من وجهة نظر شخصية) سوف لا تكون ذات قيمة لقاعدة البيانات متعددة الوصلات.

عيـوب الملف متعدد الوصلات :

(١) تخزين الوصلات داخل السجل نفسه تجعل من الصعب جداً إضافة علاقة ربط واحد-لمتعدد جديدة بعد بناء قاعدة البيانات؛ ومن ثم سوف يؤدى ذلك لإعادة بناء قاعدة البيانات من البداية، بدلاً من التغيير بشكل متحرك لعكس علاقة الربط واحد - لمتعدد الجديدة.

(٢) معالجة الاستفسارات التى تتضمن مجموعات منطقية للمفتاح الثانوى. على سبيل المثال: فى حالة افتراض أن قاعدة البيانات فى الشكل رقم (٢-١٠) كانت نتيجة

استفسار لكل الطالبات "الإناث" أو فى "الصف الدراسى الأخير". أى خوارزمية algorithm معالجة هذا المطلب فى الهيكل المتعدد القوائم سوف يتطلب تعرجاً تاماً لكل من قائمة "الإناث" وقائمة الصف "الدراسى الأخير ". سجلات الطلاب الذين فى "الصف الدراسى الأخير" و "الأناث" سوف يتم تداولها فعلياً مرتين خلال تلك الخوارزمية.

ثانيًا – الملفـات المعكوســة :

الفكرة الأساسية وراء الملفات المعكوسة هى ألا تضع فى سجلات البيانات الفعلية معلومات عن مواضع التحرير الضرورية لمعالجة المفتاح الثانوى بكفاءة . وضع معلومات عن مواضع التحرير فى سجلات بيانات للملف المتعدد الوصلات يؤدى إلى اللافعالية التى تشهد فى تلك الطريقة. يُعدُ بديلاً لذلك طريقة الملف المعكوس التى يتم البحث فيها من خلال بيانات عن بيانات فى ملفات صغيرة بحيث تبقى جزءًا من سجلات البيانات الفعلية. هذه الملفات المعكوسة الصغيرة هى فى نفس تأثير الفهارس التى تحتوى على مواضع تحرير السجل النسبية لهذه السجلات التى تشارك القيم المطابقة للمفتاح الثانوى المستخدم. على سبيل المثال، الملفات المعكوسة فى الشكل رقم (٢-١١) لقاعدة البيانات الممثلة فى الشكل رقم (٢-٨) والشكل رقم (٢-٩).

شكل رقم (٢-١٠ أ) التمثيل المادى لقاعدة بيانات الطالب (المتعدد الوصلات)

	IDENTIFIER	NAME	SEX	Sex Link	CLASS	Class Link
1	34762	أحمد على	M	2	JU	5
2	37938	إبراهيم طه	M	5	SE	9
3	12387	سارة محمد	F	4	FR	8
4	27127	علا محمد	F	6	SO	6
5	93791	حسن السيد	M	8	JU	11
6	35261	آمال فهمى	F	7	SO	7
7	59795	منى كمال	F	11	SO	12
8	23719	رامى عوض	M	9	FR	10
9	64272	أحمد عادل	M	10	SE	14
1	48262	محمد السعيد	M	12	FR	13
1	58799	شيماء كامل	F	13	JU	15
1	97271	سامح السعيد	M	15	SO	Null
1	59143	رانيا محمد	F	14	FR	Null
1	87927	دينا مدحت	F	16	SE	16
1	28098	شريف محمد	M	Null	JU	Null
1	47819	نانسى نجيب	F	Null	SE	Null

Male-Head-Pointer = 1 SO-Head-Pointer = 4 Female-Head-Pointer = 3

JU-Head-Pointer = 1 FR-Head-Pointer = 3 SE-Head-Pointer = 2

Male → أحمد على → إبراهيم طه → حسن السيد → رامى عوض → أحمد عادل → محمد السعيد → سامح السعيد → شريف محمد

Female → سارة أحمد → علاء أحمد → أمال فهمى → منى كمال → شيماء كامل → رانيا محمد → دينا مدحت → نانسى نجيب

FR → سارة محمد → علا محمد → محمد السعيد → شيماء كامل

SO → علاء محمد → أمال فهمى → منى كمال → سامح السعيد

Ju → أحمد على → حسن السيد → شيماء كامل → شريف محمد

SE → إبراهيم طه → أحمد عادل → دينا مدحت → نانسى نجيب

شكل رقم (٢-١٠ب) التمثيل المنطقى لقاعدة بيانات الطالب (متعدد الوصلات)

إن الاستفسار بالمفتاح الثانوى يؤدى إلى البحث فى الملف المعكوس لقيمة المفتاح الثانوى الخاصة. بمجرد إيجاد القيمة موضوع البحث سوف توضع قائمة كل مواضع التحرير فى ملف البيانات الفعلى ، حيث يتم البحث عن السجلات التى تحقق الاستفسار. ومن ثم الملفات المعكوسة تعكس مشكلة مثل البحث عن كل الطلاب الذين هم "بالصف الثالث" فى مشكلة البحث عن أصغر ملف لقيمة المفتاح "الصف الثالث" ؛ ومن ثم تبسيط تداول كل السجلات التى يبحث عن ترابطها مع قيمة ذلك المفتاح . البحث فى الملف المعكوس عن قيمة معينة للمفتاح يتم بكفاءة عالية؛ لأن الملف يحتوى على مواضع سجلات نسبية فقط وليست بيانات حقيقية. ومن ثم الملفات المعكوسة نفسها هى صغيرة ومع ذلك تسمح بمواضع تحرير ضخمة . طريقة الملف المعكوس تعيد حل مشكلة اللافعالية المذكورة فى الملف المتعدد الوصلات ، فيما يلى :

مزايا الملف المعكوس :

(١) إضافة ملف معكوس لعلاقة الربط واحد-لمتعدد بعد بناء قاعدة البيانات، تتطلب فقط المرور التتابعى خلال سجلات البيانات الفعلية لبناء ملف معكوس مناسب . حيث إن سجلات البيانات الفعلية لا تحتوى على معلومات عن مواضع التحرير، وليس هناك حاجة لإعادة البنـاء.

(٢) مشكلة تداول سجلات معينة مرتين لبعض الاستفسارات المنطقية لقواعد البيانات يتم تخفيفها بحجة الحاجة فقط للتطلع على الملف المعكوس لإيجاد السجلات الملائمة.

شكل رقم (٢-١١) الملفات المعكوسة لقاعدة بيانات الطالب شكل (٢ – ٨)

ملف معكوس لحقل النوع sex

القيمة	قيم مواضيع تحرير السجلات
M	١ ٢ ٥ ٨ ٩ ١٠ ١٢ ١٥
F	٣ ٤ ٦ ٧ ١١ ١٤ ١٦

ملف معكوس لحقل السنة الدراسية Class

القيمة	قيم مواضيع تحرير السجلات
FR	٣ ٨ ١٠ ١٣
SO	٤ ٦ ٧ ١٢
JU	١ ٥ ١١ ١٥
SE	٢ ٩ ١٤ ١٦

عيوب الملف المعكوس :

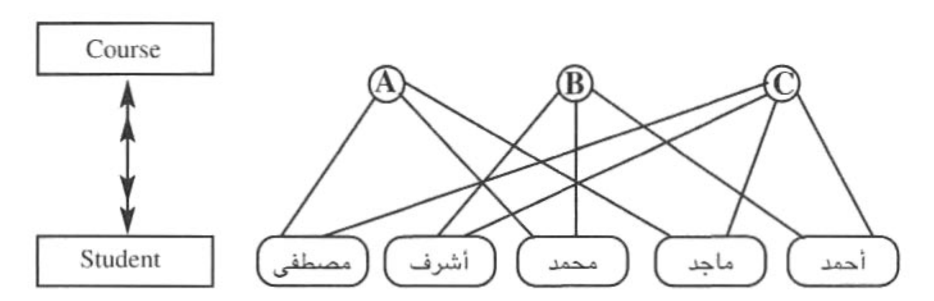
تكمن عيوب الملف المعكوس بشكل مبدئى فى التعقيدات المضافة التى يقدمها . حيث إن كل ملف معكوس يضيف ملفًا آخر لقاعدة البيانات . على سبيل المثال مشكلة حذف سجل له مفتاح أساسى يعادل مدخلاً محدداً . ولتنفيذ ذلك يجب :

- البحث فى ملف البيانات الفعلى عن طريق المفتاح الأساسى.
- تحديد قيم هذا السجل تتبنى مختلف حقول المفتاح الثانوى.
- تحفظ هذه القيم وموضع تحرير السجل لسجل البيانات الفعلى.
- لكل ملف معكوس يتم البحث عن قيمة المفتاح وموضع تحرير السجل فى الخطوة السابقة ثم حذف هذا السجل بشكل ملائم من قائمة مواضع تحرير السجلات. على الرغم من ذلك فإن هذه العملية قد تحدث بسرعة عالية لو أن سياسة البحث وتقنيات تمثيل القائمة تمت باختيار ذكى، إلا أنه لازال الحذف من الملف المعكوس يعتبر أكثر تعقيداً من الحذف من القوائم متعددة الوصلات.

ب- علاقة الربط متعدد - لمتعدد :

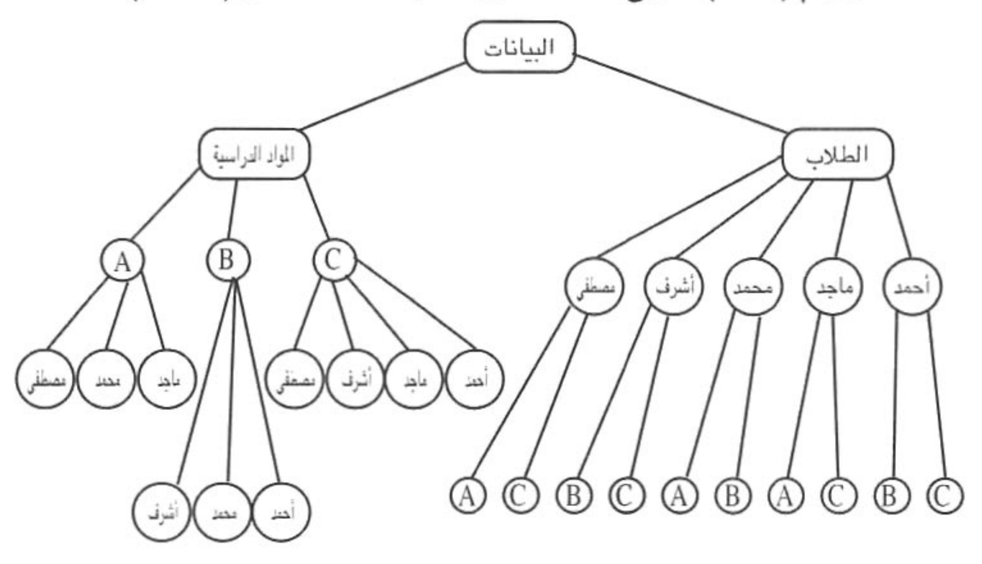
فى حالة علاقات الربط واحد - لمتعدد التى تعمل فى كلا الاتجاهين تسمى علاقة ربط متعدد - لمتعدد أو شبكة معقدة. على سبيل المثال : تعرض الجامعة فرصاً لدراسة المناهج الدراسية A . B . C : للطلاب أحمد ومحمد وماجد وأشرف ومصطفى كما هو موضح بالشكل رقم (٢ - ١٢) الأسهم ذات الرؤوس المزدوجة التى تعمل فى كلا الاتجاهين.

شكل رقم (٢-١٢) الشبكة المعقدة للفرص الدراسية والطلاب



طريقة قواعد البيانات لتمثيل علاقة الربط متعدد - لمتعدد مثلما هو مبين فى الشكل رقم (٢-١٢) هى أن تتفكك أولاً إلى فروع أشجار ممتدة Spanning Forest of trees كما هو موضح بالشكل رقم (٢-١٣) وهى مجموعة من علاقات الربط واحد - لمتعدد ، كل منها يمكن أن يتم تمثيله بإحدى التقنيات السابق ذكرها .

شكل رقم (٢-١٣) الفروع الممتدة لشجرة الشبكة المعقدة لشكل (٢ - ١٢)



وبالتركيز على الفرع الممتد Spanning forest فى الشكل رقم (٢-١٣) نجد أن :

(١) إعادة تكرار لما يعرف عن الفروع الممتدة بعلاقات الربط واحد - لمتعدد. أى أن فرصة دراسة منهج دراسى معين أو اسم طالب معين يظهر فى أكثر من رأس node فى الفرع الممتد ، وذلك المنهج الدراسى أو الطالب يتم تخزينة فعلياً فى أكثر من موقع فى قاعدة البيانات. ولكن الفرع الممتد يمثل فقط الترتيب المنطقى للبيانات فى قاعدة البيانات وليس الترتيب المادى.

(٢) على الرغم من أن بيئة قاعدة البيانات تتجنب ذكر الاتجاه ، فإن التطبيق الفعلى للفروع الممتدة للأشجار خلال تقنية الملف المتعدد الوصلات أو الملف المعكوس هى فى الحقيقة تكافئ وظيفياً تمثيل المصفوفة التبديلية Spare matrix للمصفوفة التجاورية Adjacency Matrix التى تسمح بتداول الاتجاه بواسطة الصف والعمود كما هو موضح بالشكل رقم (٢-١٤). ومن ثم فى أساليب عديدة نجد أن إدارة قاعدة البيانات هى فى الحقيقية تطبيق لهياكل البيانات.

شكل رقم (٢ - ١٤) المصفوفة التبديلية للشبكة المعقدة لشكل (٢ - ١٢)

	مصطفى	أشرف	ماجد	محمد	أحمد
A	1	0	1	1	0
B	0	1	1	0	1
C	1	1	0	1	1

تمثيل علاقات الربط لقواعد البيانات التقليدية :

أ - النموذج الهرمى :

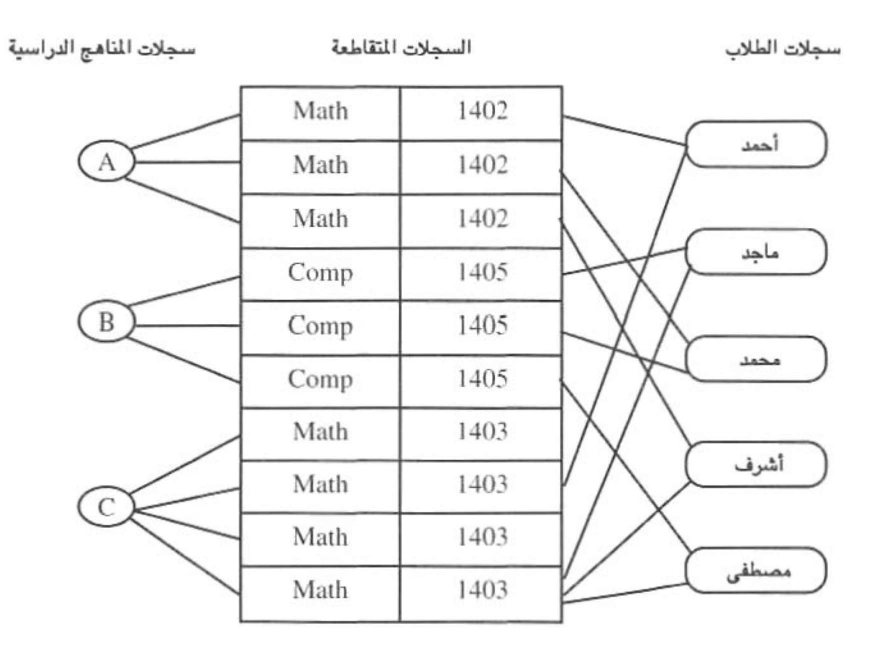
كتلة البناء الأساسية المتوافرة لمصمم قاعدة البيانات الذى يستعمل النموذج الهرمى هى الشجرة واحد - لمتعدد . وتلقائياً الملفات المتعددة الوصلات والمعكوسة الضرورية يتم بناؤها وتحفظ بواسطة نظام إدارة قواعد البيانات. ومع ذلك، على المصمم الذى يستعمل النموذج الهرمى ويرغب فى أن يبنى علاقات ربط لقاعدة بيانات

معقدة (متعدد – لمتعدد) ، أن يفكك هذه العلاقات إلى العديد من علاقات الربط واحد – لمتعدد كما هو مبين فى شكل رقم (٢–١٣). بالاضافة إلى قدرة المصمم على أن يصف أيًا من الحوادث (القيم) occurrences المتعددة للحقل فى مثل هذا التفكيك التى ينبغى أن تكون حوادث مادية ومؤشرات منطقية. ونظم إدارة قواعد البيانات بصفة عامة تستعمل الملفات متعددة الوصلات والمعكوسة التى يمكن أن تمثل مباشرة علاقات الربط واحد – لمتعدد. النموذج الهرمى يلزم مصممه أن يعمل بأحكام فى حدود علاقة الربط واحد – لمتعدد. الاستقلال الحقيقى فقط بين النموذج الهرمى وتطبيقة الضمنى هو أنه ليس من المهم للنموذج أن يستعمل سواء الملفات المتعددة الوصلات أو الملفات المعكوسة.

ب– النموذج الشبكى :

النموذج التشاورى للغة نظام البيانات CODASYL يستخدم لتوضيح الفئة Set لوصف علاقات الربط واحد – لمتعدد وهى مناظرة للشجرة التى سبق وتم توضيحها . وبتوصيف الفئة للنموذج التشاورى للغة نظام البيانات، يكون المصمم قادرًا على توصيف علاقة الربط واحد – لمتعدد والتى تبنى وتحفظ بواسطة نظم إدارة قواعد البيانات . وهذا يعنى أن فئات النموذج التشاورى للغة نظام البيانات لا تزال بشكل ضرورى شجرة هرمية . ولكن الاختلاف المفاهيمى الأساسى فيما بينها هو أن النموذج التشاورى للغة نظام البيانات لا يلزم المصمم التعامل مع الأعضاء (الأبناء) members على تواجدهم مرتين داخل قاعدة البيانات (إما ماديًا أو منطقيًا) عندما توصف علاقة الربط واحد – لمتعدد فى الرسم للنموذج الهرمى الذى يلزم المصمم أن يصف علاقة الربط واحد – لمتعدد فى حدود فرعه الممتد Spanning forest. فى حين يسمح النموذج التشاورى للغة نظام البيانات للمصمم الذى يعرف علاقة الربط واحد – لمتعدد (شبكة بسيطة) للعمل مباشرة وليس الالتزام بالفرع الممتد. ويسمح النموذج التشاورى للغة نظام البيانات للمصمم بتعريف السجلات المتقاطعة كوصلة زائفة بين السجلات فى حالة علاقة الربط متعدد – لمتعدد. على سبيل المثال: الشبكة المعقدة المبنية فى الشكل رقم (٢–١٢) والتى يمكن توضيحها فى حدود تقاطع السجلات المبينة فى الشكل رقم (٢–١٥) والتى تمثل الحوادث (القيم) المنطقية فقط.

شكل رقم (٢-١٥) يوضح السجلات المتقاطعة لتعريف الشبكة المعقدة (علاقة الربط متعدد - لمتعدد)



النموذج العلاقى :

ويبين الشكل رقم (٢-١٦) مثالاً بسيطًا للنموذج العلاقى، فيه كل صف يمثل سجل الطالب (قيم مرتبة) والأعمدة (الخصائص) تمثل حقول الاسم Name، النوع Sex والصف الدراسى Class.

شكل رقم (٢-١٦) الجدول العلاقى لسجل الطالب (القيم المرتبة للطالب)

Name	Sex	Class
خالد نيازى	M	FR
جيهان فؤاد	F	SO
هانى هلال	M	SO

تأتى قوة أسلوب النمذجة العلاقية من حقيقة أن مصمم قاعدة البيانات بفرض أنه يتصرف بحرية فى معرفة الجداول العلاقية الخاصة بتوصيف علاقات ربط معينة داخل قاعدة البيانات. ومثالاً على كيفية استعمال جدول علاقى لعكس علاقة الربط واحد - لمتعدد فى قاعدة البيانات، نأخذ على وجه الاعتبارعلاقة الربط طالب - منهج دراسى Student-Course فى الشكل رقم (٢-١٧) وهو يوضح علاقة الربط واحد - لمتعدد التى تظهر فى الشكل رقم (٢-١٢). يبين الشكل رقم (٢-١٨) الموضح للشكل رقم (٢-١٢) مثالاً على كيفية استعمال الجداول العلاقية لتعريف علاقات الربط المعقدة متعدد - لمتعدد.

شكل رقم (٢-١٧) لعلاقة ربط واحد - لمتعدد الموضحة بشكل (٢ - ١٢)

Course	Name
BUS44A	خالد نيازى
MAT44A	خالد نيازى
PHI33A	خالد نيازى
CPS11B	خالد نيازى
MAT11C	جيهان فؤاد
ENG22D	جيهان فؤاد
PHI77B	هانى هلال
PSY33A	هانى هلال
COM99A	هانى هلال
CPS33B	هانى هلال
ENG22B	هانى هلال
MAT33A	هانى هلال

شكل رقم (٢-١٨) لعلاقة ربط متعدد - لمتعدد الموضحة بشكل (٢ - ١٢)

Student	Course
ماجد	A
محمد	A
مصطفى	A
أحمد	B
محمد	B
أشرف	B
أحمد	C
ماجد	C
أشرف	C
مصطفى	C

الهوامش :

1. [NAVATHE, 1992], Shamkant B. Navathe, 'Evolution of Data Modeling for Databases', **Communications of the ACM**, Vol. 35, No. 9, September 1992.

2. [ELMASRI, 1989], Ramez Elmasri and Shamkant B. Navathe, **Fundamentals of Database Systems**, Benjamin/Cummings, Redwood City, Calif., 1989.

3. [DAVIES, 1992], P. Beynon Davies, 'Entity Models to Object Models: Object-Oriented Analysis and Database Design', **Information and Software Technology**, Vol.34, Number 4, April 1992.

4. [RIAD, 1993], Mokhtar Boshra Riad and Halim Habib, 'A System for Conversion Between Hierarchic Network, and Relational Database Models', The Egyptian Computer Science Journal, July 1993.

5. [GRANT, 1987], John Grant, **Logical Introduction to Databases**', Harcourt Brace Jovanovich, Publishers and its subsidiary, Academic Press, 1987.

6. [NAPS, 1986], Thomas L. Naps, and Bhagat Singh, **Introduction to Data Structures with Pascal**, West Publishing Company, 1986.

7. [Connolly 1996], Thomas. M. Connolly and Carolyn E. Begg, **Database Systems**, Addison-Wesley Publishing Co., Inc., 1996.

الفصل الثالث

نماذج البيانات التبحرية

مقدمة :

تشتمل نماذج البيانات التبحرية على كل من النموذج الهرمى والنموذج الشبكى. وقد اعتمد الشكل التنفيذى لنموذج البيانات الهرمى على نظام إدارة المعلومات IMS ، فى حين اعتمد الشكل التنفيذى لنموذج البيانات الشبكى على النموذج التشاورى للغة نظام البيانات CODASYL. وسوف يتم التطرق إلى النقاط التالية خلال هذا الفصل:

نموذج قاعدة البيانات الهرمية :

يأخذ الرسم التخطيطى لهيكل البيانات في نموذج قاعدة البيانات الهرمية شكل مجموعة مرتبة من الأشجار الفرعية، وتتكون كل شجرة فرعية من نوع سجل واحد يمثل أصلها. وتعتمد على هذا الأصل مجموعة مرتبة تتكون من صفر أو أكثر من أنوا ع السجلات للمستوى الأقل. وترتبط أصول الأشجار الفرعية بنوع سجل واحد يشكل أصل النظام.

هياكل البيانات الهرمية فى نظام إدارة المعلومات :

سوف يتم التطرق لبعض الأشكال التنفيذية لنظم قواعد البيانات الهرمية عامة، ولنظام إدارة المعلومات IMS خاصة. حيث يتم توضيح تبحرات قاعدة البيانات الهرمية طبقًا لقاعدة التبحر حسب الترتيب المسبق.

توصيف البيانات فى نظام إدارة المعلومات :

يتم توصيف البيانات فى نظام إدارة المعلومات داخل المخطط المفاهيمى. وذلك من خلال توصيف قاعدة البيانات للشجرة الرئيسية والأشجار الفرعية الملحقة بها. كما سيتم توضيح كيفية تخزين كل من نوع السجل ونوع الحقل، بالإضافة إلى بداية موقع كل حقل داخل السجل، بجانب الإشارة إلى الطرق المختلفة لتخزين قاعدة البيانات الهرمية والسمات التى تتميز بها. بالإضافة إلى توضيح قواعد الحفاظ على سلامة البيانات بالنموذج الهرمى.

نموذج قاعدة البيانات الشبكية :

سوف يتم توضيح مكونات النموذج الشبكى فى هذا الجزء. حيث تتكون قواعد البيانات الشبكية من مجموعتين. إحدى المجموعتين خاصة بأنواع السجلات ، والمجموعة الأخرى خاصة بالوصلات.

هياكل البيانات الشبكية فى النموذج التشاورى للغة نظام البيانات :

سوف يتم التركيز فى هذا الجزء على معنى مصطلح كلمة فئة الذى تم استخدامه فى النموذج التشاورى للغة نظام البيانات. ويعني هذا المصطلح الوصلة التى تربط بين نوعى سجلين مختلفيين.

بعض مفاهيم النموذج التشاورى للغة نظام البيانات :

هناك العديد من المفاهيم التى ينبغى التطرق لها لفهم هذا النموذج مثل: نوع السجل، حادثة السجل، القائمة المتصلة، والفئة المنفردة.

لغة تعريف البيانات :

تمثل لغة تعريف البيانات اللغة الرئيسية لقواعد البيانات الشبكية. وتتركب هذه اللغة من ثلاثة لغات فرعية يتم استخدامها لتعريف التخطيط والمخططات الفرعية ومعالجة البيانات.

وسوف يتم توضيح خيارات كل من عبارات الإضافة والاحتفاظ. سواء كانت الإضافة هيكلية ، تلقائية أم يدوية. كذلك سواء كان الاحتفاظ ثابتًا، إجباريًا أم اختياريًا.

تبحر قاعدة البيانات :

يتمثل التبحر فى مرور برنامج قاعدة البيانات خلال سجلات الملف عن طريق الوصلات للحصول على النتائج المطلوبة. وسوف يتم التطرق إلى سمات قواعد البيانات الشبكية وكيفية الحفاظ على سلامة البيانات بالنموذج الشبكى.

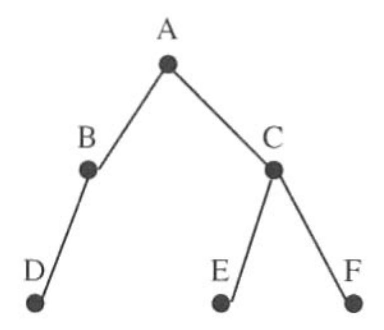
معالم نماذج البيانات التبحرية ومحدودياتها :

سوف يتم التطرق لمعالم نماذج البيانات التبحرية بشكل مبسط إلى جانب محدودية هذه النماذج وما تتضمنه من متاعب أثناء البرمجة وصعوبة عند الاستعلام.

نموذج قاعدة البيانات الهرمية :

تمثل الرؤوس nodes فى الرسم التخطيطى لهيكل البيانات فى النموذج الهرمى الملفات Files ، والتى يستخدم لها عادة فى النظم التبحرية مرادفًا لفظيًا هو عبارة عن أنواع السجلات ، فى حين تمثل الوصلات edges علاقات الربط واحد - لمتعدد. ويسمح النموذج الهرمى للرسم التخطيطى لهيكل البيانات بأن نأخذ شكل شجرة كما هو مبين بالشكل رقم (٣-١). ويتكون النموذج الهرمى لقاعدة البيانات من مجموعة مرتبة من الأشجار. نوع الشجرة الواحدة يتكون من نوع سجل أصل واحد one root مع مجموعة مرتبة تتكون من صفر أو أكثر من أنواع السجلات للمستوى الأقل معتمدة على الأشجار الفرعية. وتتكون كل شجرة فرعية من نوع سجل واحد فقط يمثل أصلها مع مجموعة مرتبة تتكون من صفر أو أكثر من أنواع السجلات للمستوى الأقل معتمدة على تلك الشجرة الفرعية وهكذا كما هو مبين فى المثال (٣-١). وأحد الأسباب المهمة فى نجاح النموذج الهرمى هو إمكانية وضع البيانات بطريقة طبيعية [١]،[٢].

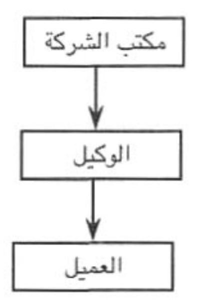
شكل رقم (٣ - ١) يمثل الرسم الهرمى (الرؤوس والوصلات)



مثال (٣-١) :

يبين الشكل رقم (٣-٢) الرسم التخطيطى لهياكل البيانات والتى توضح وجود البيانات فى شكل تتابعى لعلاقات ربط واحد - لمتعدد لشركة تأمين افتراضية Insurance Company.

شكل رقم (٣ -٢) الرسم التخطيطى لأنواع سجلات شركة التأمين الافتراضية (علاقة ربط واحد - لمتعدد)



وتتكون قاعدة بيانات شركة التأمين الافتراضية من ثلاثة أنواع من السجلات (أى من ثلاثة ملفات) هى : نوع سجل "مكتب الشركة" OFFICE، ويتكون من ثلاثة حقول هى : عنوان المكتب OAddr ، رقم التليفون TelNo ، ومدير المكتب Manager. ونوع سجل "الوكيل" AGENT، ويتكون أيضًا من ثلاثة حقول هى : اسم الوكيل AName، وعنوان الوكيل AAddr، والعمولة Comm. أما نوع سجل "العميل" CLIENT ، فيتكون من أربعة حقول هى : اسم العميل CName ، وعنوان العميل CAddr، نوع البوليصة PType ، رقم البوليصة PNum.

ويلاحظ أن نوع سجل "المكتب" OFFICE يرتبط بنوع سجل "الوكيل" AGENT. أى أنه يحتوى على عدد من سجلات وكلاء الشركة Agents. ولكن كل وكيل مرتبط بمكتب شركة واحد ، وكل نوع سجل "وكيل" AGENT قد يرتبط بعدد من سجلات العملاء ، ولكن كل سجل عميل يتعامل مع سجل وكيل واحد.

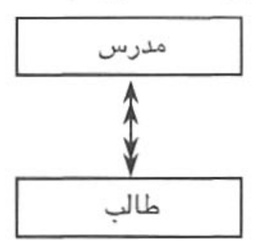
يشكل الرسم التخطيطى الشجرى لهياكل البيانات الأساسية مخطط قاعدة البيانات الهرمية . ولكن تنشأ المشكلة عند محاولة تمثيل علاقات الربط متعدد -

لمتعدد . فعند استعمال علاقتى ربط واحد - لمتعدد فى النموذج الشبكى فإن الرسم التخطيطى لهياكل البيانات لم يعد يحتفظ بالهيكل الشجرى ؛ لذا تكون هناك حيلولة فى تمثيل علاقات الربط متعدد - لمتعدد بمثل هذه الطريقة [٢].

مثال (٣-٢) :

يبين الشكل رقم (٣-٣أ) الرسم التخطيطى لعلاقة الربط متعدد - لمتعدد بين نوع سجل "المدرس" TEACHER ونوع سجل "الطالب" STUDENT والتى يمكن التعبير عنها بعبارة مدرس - طالب Teacher - Student. ويتكون نوع سجل "المدرس" TEACHER من أربعة حقول هى : رقم المدرس SS# ، اسم المدرس TName ، الدرجة الوظيفية Position، والإدارة التى ينتمى إليها Dept. ، فى حين يتكون نوع سجل الطالب من خمسة حقول هى: رقم الطالب SSNo ، اسم الطالب SName ، عنوان الطالب SAddr ، رقم تليفون الطالب TelNo ، وتصنيف الطالب Classification.

الشكل رقم (٣-٣أ) الرسم التخطيطى لعلاقة الربط (متعدد - لمتعدد) (مدرس - طالب)



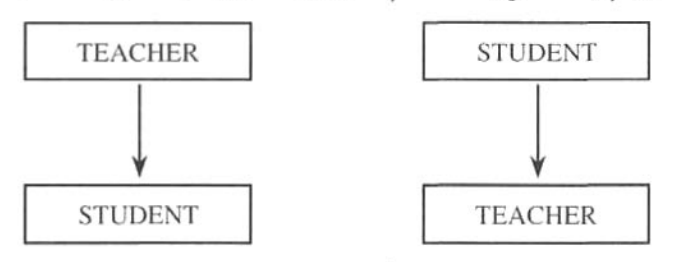
وأحد حلول مشكلة تمثيل علاقة الربط متعدد - لمتعدد يتم عن طريق استخدام شجرة واحدة لتمثيل علاقة الربط السابقة كعلاقة ربط واحد - لمتعدد. ويتم ذلك بوضع نوع سجل "المدرس" TEACHER كاب ونوع سجل "الطالب" STUDENT كابن . وبهذا التمثيل البسيط يمكن إيجاد كل سجلات الطلاب Students لسجل مدرس معين a teacher . ونظراً لأن علاقة ربط مدرس - طالب Teacher-Student هى فى الأصل علاقة ربط متعدد - لمتعدد ، فإن سجل كل طالب يجب أن يتكرر مع سجل كل مدرس Teacher حيثما يدرس المدرس الطالب كما هو موضح بالشكل رقم (٣-٣ب).

الشكل رقم (٣-٣ب) استخدام شجرة واحدة لتمثيل لعلاقة الربط متعدد - لمتعدد كعلاقة الربط (واحد - لمتعدد) (مدرس - طالب)



ولكن توجد هنا مشكلتان فى هذا التمثيل. أولاهما: تكرار البيانات بشكل كثير؛ مما يؤدى إلى تضارب البيانات. ثانيتهما: عند البحث عن كل المدرسين الذين يدرسون طالب معين تصبح عملية البحث أكثر تعقيداً؛ لأن قاعدة البيانات قد تبحث بأكملها؛ ومن ثم تؤدى إلى مشكلة اللاتماثل asymmetric. ولتصحيح مشكلة اللاتماثل فى تمثيل البيانات يمكن تقديم فكرة جديدة متضمنة شجرتين كما هو موضح فى الشكل رقم (٣-٣ج). وتتمثل فكرة استخدام شجرتان فى وضع نوع سجل "المدرس" TEACHER كأب فى شجرة وجعل نوع سجل "الطالب" STUDENT كابن فى نفس الشجرة ، فى حين أن فى الشجرة الأخرى يعتبر نوع سجل "الطالب" TUDENT هو الأب ونوع سجل "المدرس" TEACHER هو الابن.

شكل رقم (٣ - ٣ج) استخدام شجرتين لتمثيل لعلاقة الربط متعدد - لمتعدد

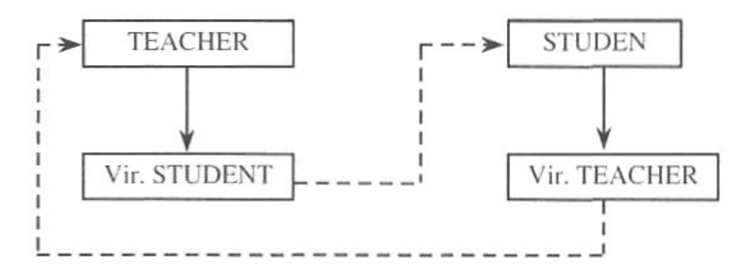


فى مثل هذه الحالة ببساطة جداً يمكن إيجاد سجلات كل المدرسين teachers الذين يدرسون لطالب معين ، وبالمثل يمكن إيجاد سجلات كل الطلاب الذين يدرسون مع مدرس معين باستخدام علاقة الربط (واحد - لمتعدد) (مدرس - طالب)

Teacher-Student فى الحالة الأولى وباستخدام علاقة الربط (واحد - لمتعدد) (طالب - مدرس) Student-Teacher فى الحالة الثانية. ولكن حالة تكرار البيانات فى هذه الحالة تصبح أكثر سوءاً عما قبل. ففى هذا الوضع الجديد ليس فقط سجل الطالب يتكرر بتكرار سجل مدرس حيث يدرس المدرس للطالب كما فى الشكل (٣-٣ب) ولكن أيضاً سجل المدرس يتكرر مع سجل كل طالب ، حيث يدرس الطالب لدى ذلك المدرس. وعيب هذه الطريقة هو تكرار البيانات بشكل سيئ.

وحل هذه المشكلة يمكننا من الحصول على أحسن تمثيل للبيانات. ويكمن هذا الحل فى استخدام الملف الافتراضى Virtual file كما هو موضح فى الشكل رقم (٣-٣د) الذى يعالج مشكلة تكرار البيانات. ويحتوى الملف الافتراضى على سجلات افتراضية virtual records. ويعتبر السجل الافتراضى مؤشراً إلى السجل الفعلى actual record. ويشار إلى هذه المؤشرات باستخدام الخطوط المنقطة ذات الرؤوس كما بالشكل رقم (٣-٣د).

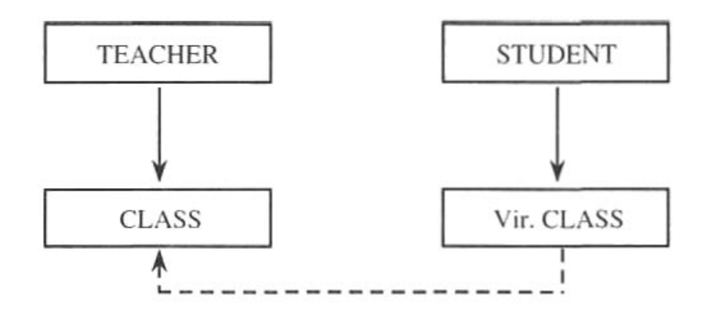
شكل رقم (٣-٣د) يوضح استخدام الملفات الافتراضية لحل مشكلة تكرار البيانات



وهنا يظهر سجل الطالب مرة واحدة فقط كسجل أب فى الشجرة الخاصة به، ويظهر سجل المدرس أيضاً مرة واحدة فقط كسجل أب فى الشجرة الخاصة به، من ثم بهذا التمثيل يمكن حل مشكلة تكرارالبيانات. وتحتفظ علاقة الربط فى هذا التمثيل بالتماثل symmetric ، حيث يناظر كل سجل حقيقى مؤشراً فى السجل الافتراضى المناظر له.

وهناك طريقتان لتمثيل علاقة الربط متعدد - لمتعدد. الطريقة الأولى: تتم باستخدام ملف الربط Connection File فى شجرة مع الملف الافتراضى المناظر له فى شجرة أخرى. وملف الربط عبارة عن ملف وسيط يستخدم لكسر جمود علاقة الربط متعدد - لمتعدد إلى علاقتى ربط كل منهما واحد - لمتعدد. وكما هو موضح بالرسم التخطيطى فى الشكل رقم (٣-٣هـ) الذى يزيد عملية التعقيد.

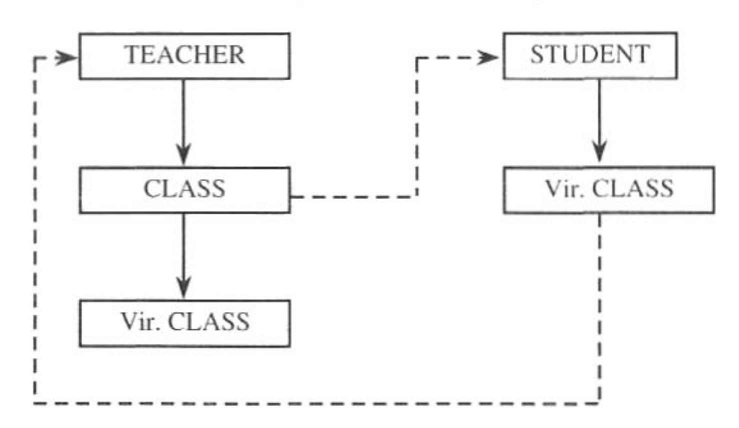
شكل رقم (٣-٣هـ) استعمال السجلات الافتراضية وسجل الربط



وتظهر هذه التعقيدات عند البحث عن كل الطلاب الذين يدرسون لدى مدرس معين فى شجرة مدرس - فصل Teacher-class. وعندئذ تفحص كل شجرة طالب - فصل افتراضى .Student-Vir. Class وتتطلب هذه العملية البحث عن سجلات الفصل الافتراضية Vir. Class لفحص مؤشرات سجلات الفصل الافتراضية لتعريف الطلاب. ويلاحظ أن الرسم التخطيطى مازال فاقد التماثل.

أما الطريقة الثانية: فهى تتضمن تعديلاً للرسم التخطيطى فى شكل رقم (٣-٣هـ) مع الاحتفاظ بمبادئة الأساسية. حيث إنه فى حالة البحث عن كل الطلاب الذين يدرسون لدى مدرس معين يمكن وضع نوع سجلات الطلاب الافتراضية تحت نوع سجلات الفصل class كما هو مبين فى الشكل التالى (٣-٣و).

شكل رقم (٣-٣و) استخدام نوع سجل الربط كنوع سجل افتراضى فى كلتا الشجرتين



وفى مثل هذه الحالة يملك كل نوع سجل فصل class (كـأب) نوع سجل طلاب افتراضى (كابن) لتعكس حقيقة علاقة الربط واحد - لمتعدد طالب - فصل Student-Class.

هياكل البيانات الهرمية فى نظام إدارة المعلومات IMS :

يعرض هذا الجزء بشكل مختصر بعض الأشكال التنفيذية لنظم قواعد البيانات الهرمية عامة ولنظام إدارة المعلومات IMS خاصة. وسوف يتم توضيح ذلك من خلال مثال شركة التأمين الافتراضية Insurance Company الذى يتكون من مستويين هرمين، مكتب الشركة - مكتب الوكيل - العملاء OFFICE-AGENT- CLIENT. وهذا يعنى وجود شجرة مرتبطة حسب نوع سجل "المكتب" OFFICE. وهذه الشجرة تحتوى على كل سجلات الوكلاء agents. كذلك نوع سجل "الوكيل" AGENT الذى يحتوى على كل سجلات العملاء Clients. وتكون قاعدة البيانات الكاملة فى نماذج البيانات الهرمية عبارة عن مجموعة مرتبة من الأشجار الفرعية Forests.

ويمثل الشكل رقم (٣-١٤أ) شجرة من هذه المجموعة. ولتقديم مثل مبسط نفترض وجود ثلاثة وكلاء agents ، كل منهم يتضمن عدداً قليلاً من العملاء Clients. يمكن تبحر traverse هذه الأشجار بطرق مختلفة. إحداها زيادة كل سجل مرة واحدة.

وأنسب هذه التبحرات لقاعدة البيانات الهرمية التبحر حسب الترتيب المسبق Preorder Traversal. وقاعدة التبحر حسب الترتيب المسبق هى :

- زيارة نوع سجل أصل الشجرة أولاً .
- ثم زيارة كل فرع للشجرة من اليسار إلى اليمين .
- ثم تبحر كل فرع بنفس الطريقة السابقة .

ولإتمام قاعدة البيانات لابد من تبحر جميع قوائم الأشجار واحدة تلو الأخرى فى ترتيب من اليسار إلى اليمين. ويحقق التبحر حسب الترتيب المسبق التنفيذ التتابعى والمتجه باستقامة كل قاعدة البيانات. ويوضح ذلك فى الشكل رقم (٣-٤ب) الخاص بشركة التأمين الافتراضية بحيث يمكن تمثيل قاعدة البيانات بالكامل كملف تتابعى.

ويلاحظ أن هذا التطبيق ليس بالتطبيق الأمثل من الناحية العملية لو أخذ فى الاعتبار عمليات الإضافة لقاعدة البيانات، وعلى أية حال يمكن النظر إلى قاعدة البيانات مثل ملف تتابعى Sequential File يلى كل سجل فيه تلو الآخر. حيث إن أى ملف (جدول) يتم تمثيله ماديًا على وسيط التخزين بشكل تتابعى.

توصيف البيانات فى نظام إدارة المعلومات IMS :

وقد تمت الإشارة فى الجزء الخاص بمقدمة قواعد البيانات إلى الانتشار الواسع لنظام إدارة المعلومات IMS المستخدم فى النظام الهرمى. واعتبارات توصيف قاعدة البيانات فى نظام إدارة المعلومات IMS يتم داخل المخطط المفاهيمى لتوصيف قاعدة البيانات حيث إن كل شجرة لها اسم يسمى توصيف قاعدة البيانات Database Description (DBD). ويتم تخزين كل من نوع السجل ونوع الحقل فى خانات bytes ، بالإضافة إلى بداية موقع كل حقل داخل السجل يجب أن يشار إليها. وتشير كلمة جزئية SEGM داخل المخطط المفاهيمى إلى اسم الملف فى الهرم (الشجرة) وهى اختصار لكلمة Segment. ويجب أن يوضع اسم سجل الأب فى كل سجل ما عدا سجل الأصل root. ويجب أن توضع أسماء حقول Fields كل ملف فى الترتيب الذى يلى اسم السجل. السجل الأول لكل جزئية يكون متسلسل ويشار إليه بكلمة تتابع SEQ وهى اختصار لكلمة Sequentional ، وتعنى أن السجلات مرتبة على ذلك الحقل الذى يمثل

حقل التسلسل و المفتاح للجزئية. وتكتب الجزئيات باستعمال طريقة الترتيب المسبق للرسم التخطيطي لهياكل البيانات[٧].

ففي الشكل رقم (٣-٤ ج) يمثل اسم توصيف قاعدة بيانات شركة التأمين الافتراضية INSPDBD الاسم المادى لقاعدة البيانات الذى يعبر عن الاسم المختصر لشركة التأمين INS المنتهى بمقطع مختصر لعبارة توصيف قاعدة البيانات DBD وهى تتكرر مع كل شجرة فرعية. ويرجع عدم تكرار كلمة DBD إلا مرة واحدة فقط فى نهاية مقطع اسم قاعدة البيانات إلى أنه لا يوجد إلا شجرة واحدة فقط[٢].

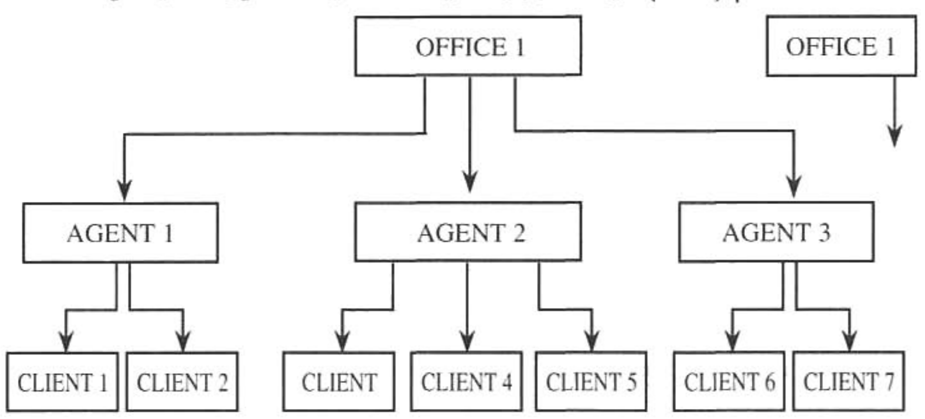
شكل رقم (٣-٤ج) جزء من المخطط المفاهيمى لتوصيف قاعدة البيانات لمثال شركة التأمين الافتراضية

```
DBD Name = INSPDBD
SEGM NAME = OFFICE ,BYTES = 43
FIELD NAME = (OAddr, SEQ) , 20 , Star = 1
FIELD NAME =TELNO , Bytes = 8, start=21
FIELD NAME = Manger , Bytes = 15 , start = 1
SEGM NAME = AGENT , Parent = OFFICE, Bytes = 39
FIELD NAME = (AName, SEQ) , 15 , Star = 1
FIELD NAME =AAddr , Bytes = 20, start = 16
FIELD NAME = Comm , Bytes = 4 , start = 36
SEGM NAME = CLIENT , Parent = AGENT, Bytes = 49
FIELD NAME = (CName, SEQ) , 15 , Star = 1
FIELD NAME =CAddr , Bytes = 20, start = 16
FIELD NAME = PType , Bytes = 1 , start = 36
FIELD NAME = PNum , Bytes = 1 , start = 36
```

ويمثل حقل عنوان مكتب الشركة OAddr حقل تتابعى ، ومكاتب الشركة مرتبة ترتيبًا تصاعديًا على ذلك الحقل الذى لا يتكرر . وحقل اسم الوكيل AName يمثل مفتاحًا لنوع سجل "الوكيل" AGENT داخل نوع سجل "المكتب" OFFICE المحدد ، وهكذا يوجد العديد من الوكلاء فى قاعدة البيانات لهم نفس الاسم ولكن ليس فى نفس

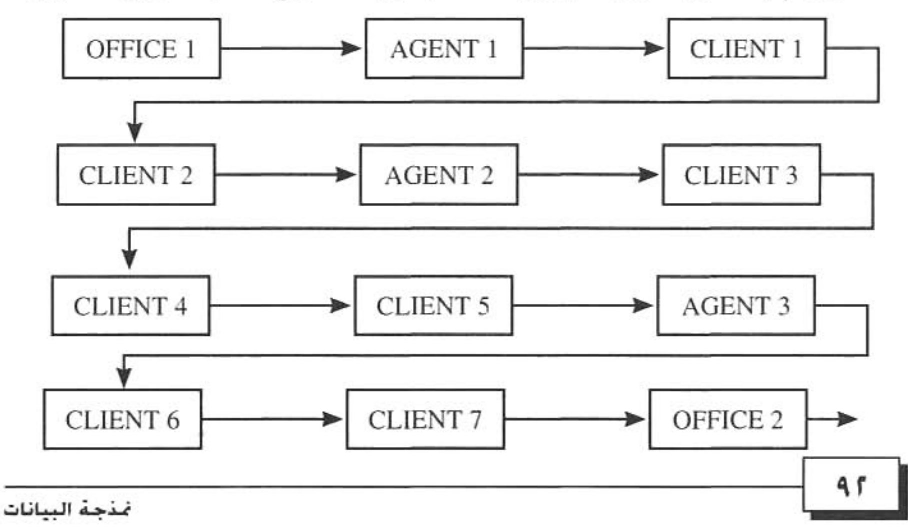
المكتب. وكذلك حقل اسم العميل CName يمثل مفتاحًا لسجلات العملاء داخل نوع سجل "المكتب" OFFICE المحدد لنوع سجل "الوكيل" AGENT حسب الهرم (الشجرة). ومن الجدير بالذكر التنويه إلى أن الشكلين رقم (٣-٤أ،ب) لا يشيران إلى قيم معينة للحقول بل لمزيد من الإيضاح تتم الإشارة إلى أرقام سجلات مكاتب الشركة ووكلاء الشركة والعملاء.

شكل رقم (٣-٤أ) هيكل شجرى لتنفيذ مثال شركة التأمين الافتراضية





شكل رقم (٣-٤ب) تطبيق تتابعى باستعمال طريقة التعرج حسب الترتيب السابق



يشير توصيف قاعدة البيانات إلى المنظور المفاهيمي . ولكن فى نظام إدارة المعلومات IMS يتعامل المستخدمون مع المنظور الخارجي الذى يعرف باسم كتلة برنامج الاتصال Program Conmumication Block (PCB). يتم الحصول بشكل أساسى على المنظور الخارجى من المنظور المفاهيمى بحذف الخانات المختلفة معه فى التوصيف. مثالاً على ذلك عند حذف سجل معين من شجرة معينة يجب حذف أبناء هذا السجل. عند حذف بعض الحقول من داخل سجل معين يمكن إعادة ترتيب الحقول المتبقية مرة أخرى. ويبين الشكل رقم (٣-٥) المنظور الخارجى لقاعدة البيانات مع ملاحظة أن السجلات والحقول يجب أن تكون موجودة فى المنظور الخارجى. من المصطلحات المستخدمة فى نظام نظام إدارة المعلومات IMS عبارة "ذات معنى" Sensitive وهى عبارة تسبق نوع السجل أو الحقل فى المنظور الخارجى. ومن الجزئيات ذات المعنى نوع سجل "المكتب" OFFICE، نوع سجل "الوكيل" AGENT ، نوع سجل "العميل" CLIENT . وهذه الجزئيات تمنع دخول أى حقل تحتويه ما لم يتم الإشارة إلى تلك الحقول أثناء التوصيف للمعالجة الاختيارية PROCessing OPTions التى تستخدم المختصر PROCOPT. وتستعمل المنظور الخارجى لأغراض إعدادات الاسترجاع فقط من خلال مجموعة من الأحرف الدلالية منها: حرف G للاحضار (Get) ، حرف I للإضافة (Insert) وحرف D للحذف (Delete) وحرف R للاستبدال (Replace). وخانة طول الحقل KEYLEN تشير إلى أقصى طول لتسلسل المفاتيح، وفى هذه الحالة يساوى (٥٠) حرفًا حيث تمثل (٢٠) لحقل عنوان المكتب OAddr ، (١٥) لحقل اسم الوكيل AName ، (١٥) لحقل اسم العميل CName.

شكل رقم (٣ - ٥) المنظور الخارجى لقاعدة بيانات شركة التأمين الافتراضية

```
PCB TYPE       = DB , DBDNAME = INSCPDED , KEYLEN=50
SENSEG  NAME = OFFICE , PROCOPT = G
SENSEG  NAME = AGENT , PARENT = OFFICE , PROCOPT = G
SENFLD NAME = ANAME , START = 1
SENFLD NAME = ADDR , START = 16
SENFLD NAME = COMM , START = 36
```

SENSEG NAME = CLIENT , PARENT = AGENT , PROCOPT = G

SENFLD NAME = CNAME , START = 1

SENFLD NAME = CADDR , START = 16

SENFLD NAME = PTYPE , START = 36

SENFLD NAME = PNUM SENFLD NAME , START = 37

وفيما يلى بعض مفاهيم هياكل التخزين بنظام إدارة المعلومات IMS. حيث تبين هياكل تخزين نظام إدارة المعلومات IMS طرقًا مختلفة لتخزين قاعدة البيانات الهرمية. وقد وفر نظام إدارة المعلومات IMS هياكل التخزين التالية:

* طريقة التداول التتابعى الهرمى

Hierarchical Sequantial Access Method (HSAM)

* طريقة التداول التتابعى الهرمى البسيط

Simple Hierarchical Sequantial Access Method (SHSAM)

* طريقة التداول التتابعى المفهرس الهرمى

Hierarchical Indexed Sequantial Access Method (HISAM)

* طريقة التداول التتابعى المفهرس الهرمى البسيط

Simple Hierarchical Indexead Sequational Access Method (SHISAM)

* طريقة التداول التتابعى التعميمى

Generalized Sequational Access Method (GSAM)

* طريقة التداول المباشر الهرمى

Hierarchical Direct Access Method (HDAM)

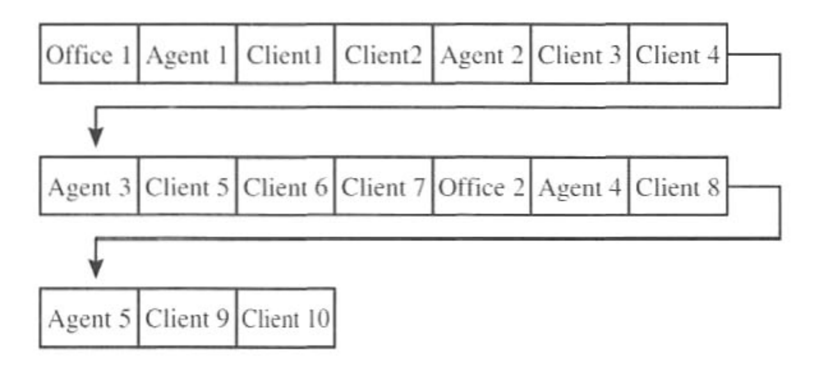
* طريقة التداول المباشر المفهرس الهرمى

Hierarchical Indexed Direct Access Method (HIAM)

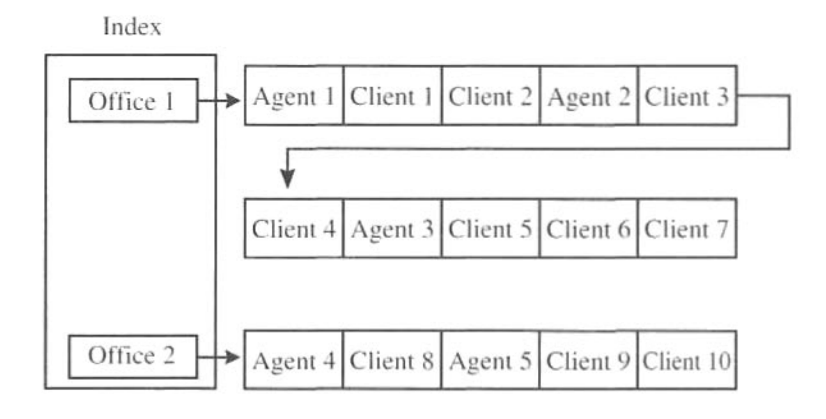
فى طرق التداول التتابعى ، توضع سجلات الشجرة بشكل مادى فى ترتيب تسلسلى فى حين طرق التداول المباشر توزع السجلات ماديًا وتوصل بالمؤشرات. وينعكس الفرق بين طريقة التداول التتابعى الهرمى HSAM . وطريقة التداول التتابعى المفهرس الهرمى HISAM فى التمييز بين الملفات التتابعية والملفات التتابعية المفهرسة. ويوضح الشكل رقم (٣-٦ أ ، ب) هذا التميز لمثال شركة التأمين الافتراضية. على سبيل المثال لو فرض أنه يوجد مكتبان للشركة: الأول له ثلاثة وكلاء فى حين أن الثانى له وكيلان وكل وكيل له واحد ، اثنين أو ثلاثة عملاء. ومن الجدير بالذكر التنويه إلى أن

الشكلين رقم (٣-٦ أ ، ب) لا يشيران إلى قيم معينة للحقول بل لمزيد من الإيضاح سوف تتم الإشارة الى أرقام سجلات مكاتب الشركة ووكلاء الشركة والعملاء.

شكل رقم (٣-٦أ) طريقة تداول التتابع الهرمى HSAM لمثال شركة التأمين الافتراضية



شكل رقم (٣-٦ب) طريقة تداول التتابع المفهرس الهرمى HISAM لمثال شركة التأمين الافتراضية



يتم استعمال المؤشرات المضمنة فى طريقة تداول التتابع الهرمى HSAM ، وطريقة تداول التتابع المفهرس الهرمى HDAM لإيجاد العنصر التالى. ويوجد طريقتان مختلفتان لتجهيز المؤشرات :

- **الطريقة الهرمية** hierachied method :

وقد استخدم نظام إدارة المعلومات IMS المصطلح الهرمى HIER للطريقة الهرمية التى تستعمل المؤشرات من خلال القائمة المتصلة Linked list للمدخلات باستعمال طريقة الترتيب المسبق كما هو موضح فى الشكل رقم (٣-٧أ).

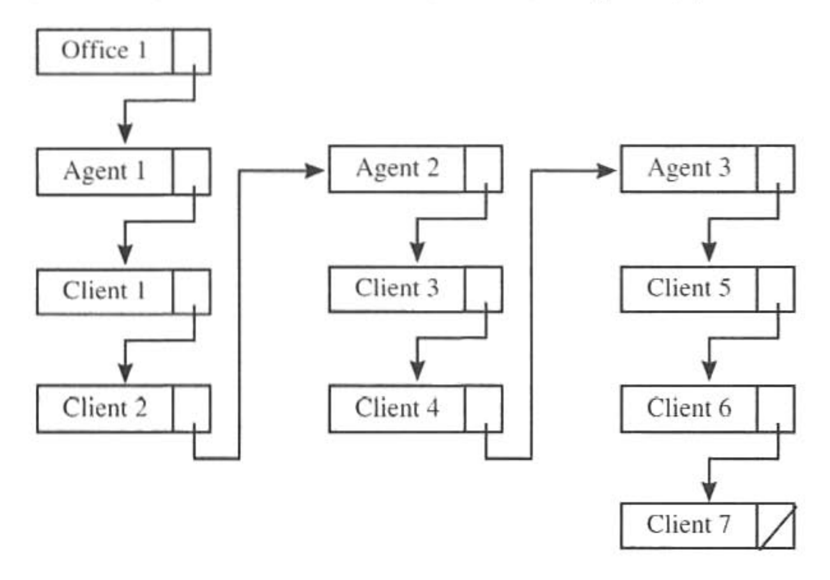
- **طريقة التوائم** Sibling-child method :

وقد استخدم نظام إدارة المعلومات IMS مصطلح التوءم TWIN لطريقة التوائم وفيها يتم توجيه المؤشرات من سجل الأب إلى سجل الابن الأول وإلى سجل الأخ التالى له. وهذه الطريقة تتضمن العديد من المؤشرات ولكن تؤدى إلى التداول السريع لقاعدة البيانات كما هو موضح فى الشكل رقم (٣-٧ب).

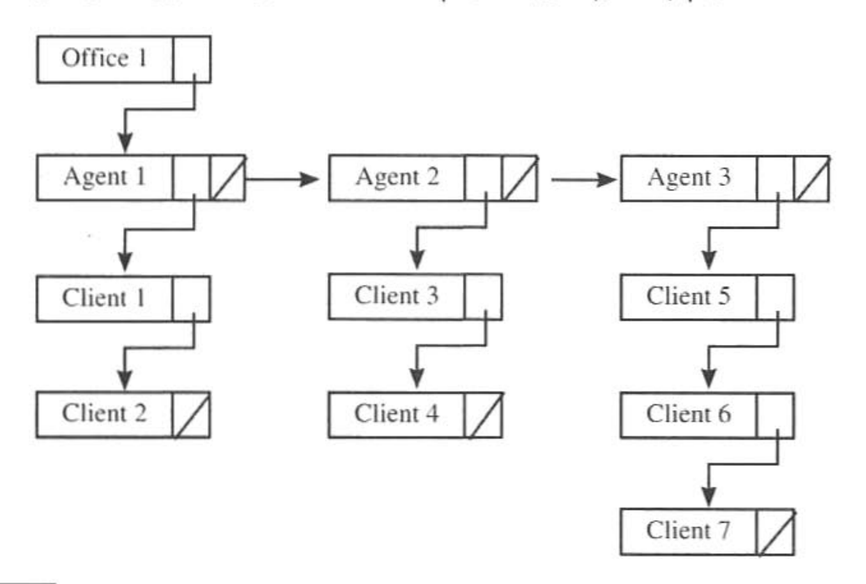
يلاحظ أن الفرق بين طريقة التداول المباشر المفهرس الهرمى HIDAM وطريقة التداول المباشر الهرمى HDAM يشابه الفرق بين طريقة التداول التتابعى المفهرس الهرمى HISAM وطريقة التداول التتابعى الهرمى HSAM. ففى طريقة التداول المباشر الهرمى HDAM يتم التداول المباشر باستخدام التفريم hashing الذى يسمح بإيجاد سجل أصل معين root بشكل سريع جداً. أما فى طريقة التداول التتابعى المفهرس الهرمى HIDAM فإن السجلات الأصلية تكون مفهرسة مثلما هو فى طريقة التداول التتابعى المفهرس الهرمى HISAM.

وهناك بعض الأساليب الخاصة بنظام إدارة المعلومات IMS ، وهى طرق معقدة جداً وإحدى هذه الطرق هى الفهرسة الثانوية Secondary Indexing ، وهذه الطريقة يمكن أن تستخدم لفهرسة حقل معين وهو ليس حقل مفتاح أو حقل تسلسل. وهذه الطريقة تسمح بالتداول السريع لمؤشرات قاعدة البيانات بالمبنية على قيم معينة فى حقل معين.

شكل رقم (٣-٧ أ) الطريقة الهرمية HIER لمثال شركة التأمين الافتراضية



رقم (٣-٧ب) طريقة التوائم TWIN لمثال شركة التأمين الافتراضية



السمات المميزة لقواعد البيانات الهرمية :

١- تحتوى شجرة قاعدة البيانات الهرمية على أصل root يمثل نوع سجل record type الأب للسجلات التالية.

٢- يوجد لنوع سجل الأصل root record عدد من المستويات التى تليه وقد يكون عددها صفراً أو أكثر مشكلاً الشجرة الفرعية ونوع سجل .

٣- الابن يمثل السجل السابق له فى المستوى الأعلى لنوع سجل الأب، والذى يعتمد عليه نوع سجل الابن. وهكذا حتى ان السجل الابن يصبح سجل أب للسجلات التالية له فى المستوى الأقل والمعتمدة عليه.

٤- تمثل العلاقة بين نوع سجل الابن ونوع سجل الأب بوصلة تسمى الأب-الأبن Par-ent-child وليس له مفتاح خارجى Forpign key كما فى النموذج العلاقى الذى سيرد شرحه.

٥- كل سجلات الأبناء التى لها سجل أب واحد تسمى توائم .

سلامة البيانات بالنموذج الهرمى Integrity part of Hierarchical Model :

تتم سلامة البيانات تلقائيًا من خلال الدعم للأشكال المحددة لقيود السلامة المرجعية referential integrity باتباع القواعد التالية :

١- لا يوجد سجل ابن بدون سجل أب له.

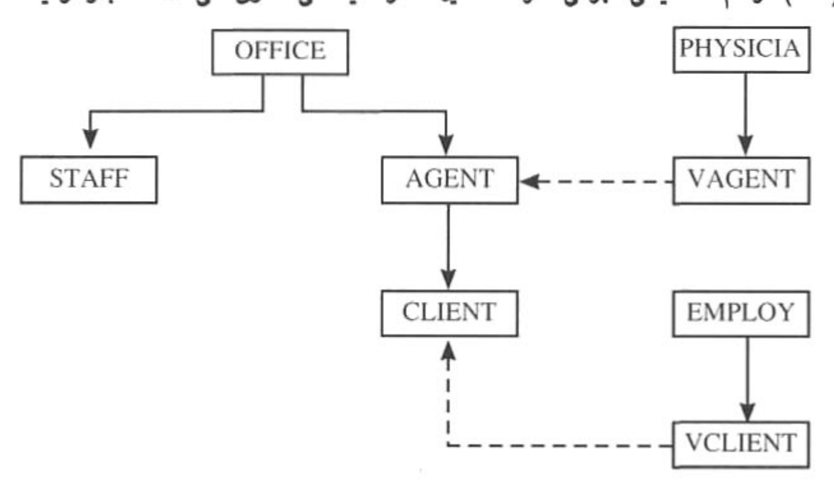
٢- لا يمكن إضافة سجل ابن دون وجود سجل الأب الخاص به.

توضيحًا لذلك عندما تأخذ هياكل البيانات شكل شجرة فإن الرسم التخطيطى يشكل الأساس لمخطط قاعدة البيانات الهرمية، ولكن المشكلة تكمن فى تمثيل علاقة الربط متعدد - لمتعدد. عندما يكون هناك علاقتان كل منهما واحد - لمتعدد فى النموذج الهرمى ، فإن شكل هياكل البيانات لم يعد موجوداً فى هيكل شجرى . لذلك لا تستعمل علاقات الربط الممثلة متعدد - لمتعدد.

مثال شركة التأمين الافتراضية المبينة فى الشكل رقم (٣- ٨) يحتوى على ثلاثة أشجار فرعية تبين العلاقة بين الأب وأبنائه وعلاقة الربط هنا هى واحد - لمتعدد لكل أب - ابن وذلك بعد إضافة نوع سجل "الطبيب" PHYSICIANT الذى يتضمن ثلاثة حقول هى: اسم الطبيب PName كحقل وحيد ، عنوان الطبيب Paddr ورقم التليفون TelNo. وأيضًا إضافة نوع سجل "صاحب العمل" EMPLOYER الذى يتضمن حقلين هما : اسم صاحب العمل Comp Name كحقل وحيد ، وعنوان صاحب العمل CompAddr.

فى نوع سجل "المكتب" OFFICE يحتفظ بعنوان المكتب OAddr وحيداً. أما فى نوع سجل "أعضاء الهيئة" STAFF المرتبط بنوع سجل "المكتب" OFFICE يحتفظ بحقل اسم عضو الهيئة SName وحيداً وحقل عنوان عضو الهيئة SName وحقل الدرجة الوظيفية لعضو الهيئة Position. وفى نوع سجل "وكيل" AGENT يحتفظ بحقل اسم الوكيل Aname وحقل عنوان الوكيل AAddr كمفتاح. فى نوع سجل "عميل" CLIENT يحتفظ بكل من حقل رقم البوليصة PNum وحقل اسم العميل CName وحقل عنوان العميل CAddr كمفاتيح، ويحتفظ أيضًا بحقل نوع البوليصة PType .ويحتفظ بنوع السجل الافتراضى لكل من الوكيل VAGENT و العميل VCLIENT.

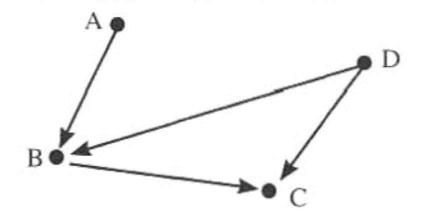
شكل رقم (٣-٨) الرسم التخطيطى الهرمى لشركة التأمين الافتراضية التى تحتوى على ثلاثة أشجار فرعية



نموذج قاعدة البيانات الشبكية Network Database Models :

تمثل الرؤوس nodes فى الرسم التخطيطى لهيكل البيانات فى النموذج الشبكى الملفات (أنواع السجلات) Files ، فى حين تمثل الوصلات edges علاقة الربط واحد - لمتعدد. ويسمح النموذج الشبكى للرسم التخطيطى لهيكل البيانات بأن يأخذ شكل رسم كما هو موضح فى الشكل رقم (٣-٩) (٢) .

شكل رقم (٣-٩) تمثيل النموذج الشبكى



تتكون قواعد البيانات الشبكية من مجموعتين، إحدى هذه المجموعات خاصة بأنواع السجلات والمجموعة الأخرى خاصة بالوصلات Links.

وتتضمن كل وصلة نوعين من أنواع السجلات . أحدها نوع سجل الأب Parent والآخر نوع سجل الابن Child. محتوى كل وصلة يتكون من حادثة (قيمة) فردية Occurrance لنوع سجل الأب مع مجموعة مرتبة من الحوادث (قيم متعددة) لنوع سجل الابن. وتعنى كلمة حادثة فى نظم قواعد البيانات الشبكية قيمة أو واقعة.

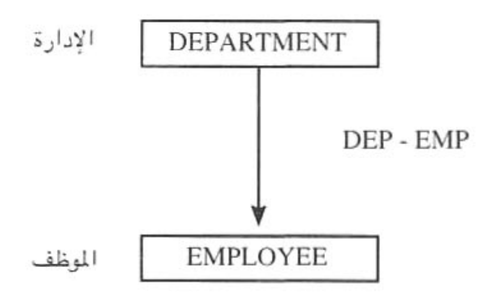
ويتم التمييز الرئيسى بين هياكل البيانات - الهرمية والشبكية عن طريق نوع سجل الابن حيث فى الهيكل الهرمى يكون لنوع سجل الابن نوع سجل أب واحد فقط فى حين أن فى الهيكل الشبكى قد يكون لنوع سجل الابن نوع سجل أب واحد أو أكثر.

هياكل البيانات الشبكية فى النموذج التشاورى للغة نظام البيانات CODASYL :

فى مصطلحات النموذج التشاورى للغة نظام البيانات CODASYL كلمة فئة set الوصلة بين كل رأسين. فى حين أن كلمة فئة فى علم الرياضيات تعنى مجموعة من القيم المرتبة (١) .

يوضح الشكل رقم (٣-١٠) فئة النموذج التشاورى للغة نظام البيانات CODASYL الذى يمثل رأسين هما ملف (نوع سجل) "الإدارة" DEPARTMENT وملف (نوع سجل) "الموظف" EMPLOYEE. فى حين تمثل الوصلة إدارة - موظف EMP - DEP فئة النموذج التشاورى للغة نظام البيانات CODASYL. يصل السهم من ملف "الإدارة" DEPERTMENT إلى ملف "الموظف" EMPLOYEE ليشير إلى أن علاقة الربط هى واحد - لمتعدد بين ملف "الإدارة" DEPARTMENT وملف "الموظف" EMPLOYEE.

شكل رقم (٣-١٠) يوضح فئة النموذج التشاورى للغة بيانات النظام CODASYL



بعض مفاهيم النموذج التشاورى للغة نظام البيانات CODASYL :

فى النموذج التشاورى للغة نظام البيانات CODASYL يعتبر ملف الأب Parent هو الأب Owner وفى هذه الحالة يعتبر الأب Owner و ملف "الإدارة" DEPARTMENT ويعتبر ملف الابن Child الملف العضو member وهو فى هذه الحالة ملف "الموظف" EMPLOYEE .

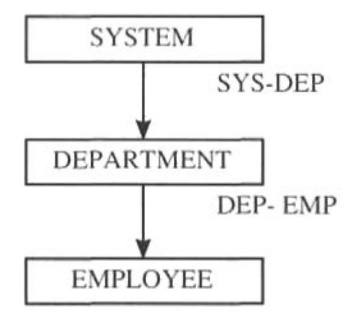
- نوع سجل النموذج التشاورى للغة نظام البيانات CODASYL مثل "الموظف" EMPLOYEE هو توصيف للملف، ونوع فئة النموذج التشاورى للغة نظام البيانات CODASYL مثل إدارة - موظف EMP ـ DEP يشير إلى الوصلة بين نوعى السجلين.
- فى حين أن حادثة occurrance السجل هى قيم السجل ، فى حين حادثة فئة النموذج التشاورى للغة نظام البيانات CODASYL تشير إلى حادثة سجل الأب وحوادث سجلات الأعضاء المناظرة له.

- القائمة المتصلة Linked List هى تتابع خطى لمجموعة عناصر لها نفس النوع ويتم الترابط فيما بينها باستخدام المؤشرات . ولكن رأس بداية القائمة المتصلة قد يكون له نوع مختلف عن نوع هذه العناصر.

يمكن تطبيق فئة النموذج التشاورى للغة نظام البيانات CODASYL السابقة إدارة – موظف DEP ـ EMP باستخدام القوائم المتصلة دائريًا مع رأس بداية ، بحيث تمثل كل قائمة منها سجلًا من ملف "الإدارة" DEPARTMENT مرتبطًا بسجلات الأعضاء لملف "الموظف" EMPLOYEE . لو كان الهيكل قائمة متصلة فرديًا Singly linked فإنه يمكن البحث داخل سجلات الموظفين من سجل إلى السجل الذى يليه فى نفس الإدارة بشكل سريع . لو كانت القائمة المتصلة مزدوجة Doubly linked فإنه من السهل الوصول إلى السجل السابق. لو وجد مؤشر إلى رأس البداية من أى عنصر، عندئذ يمكن الوصول سريعًا إلى سجل الأب لأى سجل عضو خاص. ويمكن دمج هذه الطريقة مع القوائم المتصلة المزدوجة وذلك لسرعة تداول السجلات فى قاعدة البيانات.

- الفئة المفردة Singular Set لها نظام قاعدة بيانات كنوع السجل الأب فى شكل ملف وهى يسمى "نظام" SYSTEM ، وسجلات هذا الملف لا ترتبط بآخر وتكون مترابطة معًا. كما هو وارد بالشكل رقم (٣-١١).

شكل رقم (٣-١١) يتضح فيه شكل هرمى ذو مستوى واحد مع الفئة المفردة المضافة



مثال (٣-٣) :

يسمح بحالات الهياكل اللاهرمية في النموذج الشبكي . ففي المثال الوارد بالنموذج الهرمي الخاص بالمدرس والطلاب Teacher-Student والتي تمثل علاقة الربط بين هاتين الملفين متعدد - لمتعدد يلزم إنشاء ملف ربط Connection file ، بحيث يسمح له بألا يحتوي على أية حقول. وفي هذه الحالة يكون ملف الربط هو ملف "فصل" CLASS والذي يحتوي على معلومات عن المناهج الدراسية Courses والتقديرات[٢] ويوضح ذلك بالشكل رقم (٣-١٢).

شكل رقم (٣-١٢) يوضح الرسم التخطيطي لعلاقات الربط متعدد - لمتعدد



لغة تعريف البيانات Data Difinition Language (DDL) :

لغة قواعد البيانات الشبكية Network Database Language (NDL)

هي لغة مركبة من ثلاث لغات فرعية ، هي :

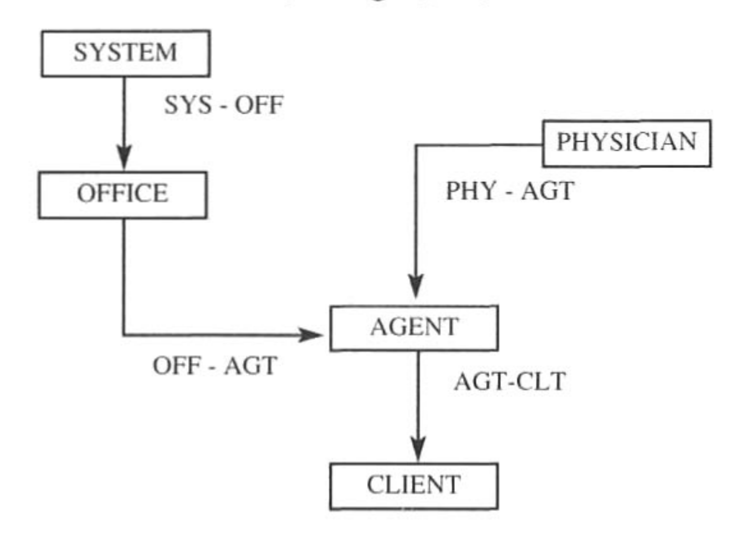
- لغة تعريف التخطيط Schema Definition Language.
- لغة تعريف المخطط الفرعي Sub-Schema Definition Language.
- لغة معالجة البيانات Data Manipulation Language.

ولغة تعريف البيانات DDL هي لغة مركبة من لغة تعريف التخطيط مع لغة تعريف المخطط الفرعي. وتستعمل لغة تعريف التخطيط لتوصيف المنظور المفاهيمي.

يبين الشكل رقم (٣-١٣) الرسم التخطيطي لهيكل بيانات شركة التأمين الافتراضية الذي تم تعديله والذي يحتوي على فئة مفردة . ويمثل الشكل رقم (٣-١٤)

عينة لمخطط التعريف لهذا المثال ، ومخطط التعريف يبدأ بتسمية المخطط . والبناء العام لهيكل مخطط التعريف يتكون من تعريف نوع السجل (الملف) أولاً ، ثم نوع فئة النموذج التشاورى للغة نظام البيانات CODASYL وتعريف كل ملف يبدأ بتسمية الملف ثم يتبعه أنواع البيانات الخاصة بحقول هذا الملف. ويمكن تعريف حقل أو مجموعة حقول لا تتكرر قيمتها كمفتاح. وهكذا بالنسبة لملف "المكتب" OFFICE يكون حقل العنوان OAddr هو المفتاح، أما بالنسبة لملف "الوكيل" AGENT فإن حقل الاسم AName وحقل العنوان AAddr يشكلا معًا المفتاح وفى ملف "العميل" CLIENT يتشكل المفتاح من حقل رقم بوليصة التأمين PNum ، وكل من حقلى اسم العميل CNane وعنوان العميل CAddr يشكلان المفتاح[٢]،[٧] .

شكل رقم (٣-١٣) الرسم التخطيطى لهيكل بيانات شركة التأمين الافتراضية الذى يحتوى على فئة مفردة



شكل رقم (٣ –١٤) عينة لمخطط التعريف

```
SCHEMA    INSURANCE-COMPANY
    RECORD OFFICE
  UNIQUE OADDR
  ITEM OADDR CHARACTER 20
  ITEM TELNO CHARACTER 10
  ITEM MANAGER CHARACTER 15
RECORD     AGENT
  UNIQUE ANAME, AADDR
  ITEM AOFFICE CHARACTER 20
  ITEM ANAME CHARACTER 15
  ITEM AADDR CHARACTER 20
  ITEM COMM SIXED 8  2
RECORD   PHYSICIAN
  ITEM PNAME CHARACTER 15
  ITEM PADDR CHARACTER 20
  ITEM TELNO CHARACTER 10
RECORD  CLIENT
  UNIQUE CNAME , CADDR
  UNIQUE PNUM
  ITEM CNAME CHARACTER 15
  ITEM CADDR CHARACTER 20
  ITEM PTYPE CHARACTER1
  ITEM PNUM CHARACTER 13
SET OFF-AGE
  OWNER OFFICE
  ORDER SORTED DUPLICATES PROHIBITED
  MEMBER AGENT
  INSERTION STRUCTURAL AGENT, AOFFICE = OADDR
```

```
   RETENTION FIXED KEY ASCENDING ANAME, AADDR
SET   PHY-AGT
  OWNER PHYSICIAN
  ORDER DEFAULT
  MEMBER AGENT
  INSERTION MANUAL
  RETENTION OPTIONAL
SET   AGENT-CLIENT
  OWNER AGENT
  ORDER FIRST
  MEMBER CLIENT
  INSERTION AUTOMATIC
  RETENTION MANDATORY
SET   ALL-OFFICES
  OWNER SYSTEM
  ORDER LAST
  MEMBER OFFICE
  INSERTION AUTOMATIC
  RETENTION FIXED
```

وكل فئة فى النموذج التشاورى للغة نظام البيانات CODOSYL يجب أن تُعطى اسمًا ، عندئذ يجب تعريف ملف الأب owner متبوعًا بعبارة ORDER التى تعرف الترتيب الذى فيه السجلات الجديدة لملف الابن والذى يسمى عضواً MEMBER ، ويضاف فى فئة النموذج التشاورى للغة نظام البيانات CODASYL ، ففى فئة مكتب- عميل Office- Agent تكون السجلات فى ملف "الوكيل" AGENT مرتبة طبقًا للمفتاح والذى يتركب من حقلى اسم العميل AName وعنوان العميل AAddr معًا . وتشير عبارة DUPLICATES PROHIBITED إلى أن محاولة تكرار قيمة المفتاح لملف "الوكيل" AGENT فى فئة مكتب – وكيل Office-Agent غير مصرح بها. وعبارة ORDER IS DEFAULT تسمح للنظام بتعريف الترتيب تلقائيًا.

أما بالنسبة لعبارات الإضافة Insertion والاحتفاظ Retention فإنه يوجد لكل منهما ثلاثة خيارات :

أولاً : بالنسبة لعبارة الإضافة فتوجد إضافة هيكلية وإضافة تلقائية وأخرى يدوية . ففى الإضافة الهيكلية واليدوية يضاف سجل العضو فى حادثة فئة النموذج التشاورى للغة نظام البيانات CODASYL المناسبة عند إضافته لقاعدة البيانات وموضح ذلك لفئة مكتب – وكيل Office-Agent.

(أ) تعنى فكرة الإضافة الهيكلية إيجاد سجل الأب عن طريق تساويه بقيمة حقل معين فى سجل العضو المضاف. مثالاً على ذلك لو تم اختيار سجل الأب بتعريف قيمة سجل المكتب Office فى سجل الوكيل Agent مع قيمة حقل عنوان المكتب OAddr فى سجل المكتب. بهذا الأسلوب يتم وضع سجل كل وكيل جديد فى حادثة فئة مكتب – وكيل Office-Agent المناسبة.

(ب) أما بخصوص الإضافة التلقائية فإنها تعنى أن سجل الأب أكثر سجل يتم تداوله حديثًا لفئة النموذج التشاورى للغة نظام البيانات CODASYL. مثال ذلك : فئة وكيل – عميل Agent-Client عندما يضاف سجل عميل جديد، فإن أباه يصبح سجل الوكيل؛ ومن ثم يكون أكثر السجلات تداولاً حديثًا.

(ج) أما حالة الإضافة اليدوية فإنها تعنى أن سجل العضو الجديد يجب أن يتم إضافته خصوصًا فى حادثة فئة النموذج التشاورى للغة نظام البيانات CO-DASYL بأية جملة . وخير مثال على ذلك هو فئة طبيب – وكيل Physician-Agent. فإنه عندما يتم إضافة سجل وكيل جديد لقاعدة البيانات ، ففى وقت إضافته لا يوضع فى أى حادثة لفئة طبيب –عميل Physician-Agent.

ثانيًا: بالنسبة لعبارة الاحتفاظ فإنه يوجد ثلاثة خيارات هى : الاحتفاظ الثابت ، والاحتفاظ الإجبارى ، والاحتفاظ الاختيارى . ويقصد بفكرة الاحتفاظ الثابت أن أى سجل للعضو بمجرد وضعه فى فئة النموذج التشاورى للغة نظام البيانات CODASYL خاصة فإنه يتم الاحتفاظ به ما لم يتم حذفه أو إعادة إضافته.

(أ) فى حالة الاحتفاظ الإجبارى فإن أى سجل للعضو بمجرد وضعة فى فئة Codasyl معينة ، فإن وجوده يكون إجبارياً فى قاعدة البيانات لو لم يتم حذفه أو أعيد إضافته ؛ لكى يكون سجل عضو لفئة النموذج التشاورى للغة نظام البيانات CODASYL لذلك النوع .

(ب) بخصوص الاحتفاظ الاختيارى فيقصد به أن وجود أى سجل كسجل عضو معين فى فئة النموذج التشاورى للغة نظام البيانات CODASYL يكون اختياراً بشكل تام – يمكن تحريكه من فئة النموذج التشاورى للغة نظام البيانات CODASYL ويظل موجوداً فى قاعدة البيانات .

(ج) يكون الاحتفاظ ثابتًا كما هو موضح فى حالة فئة مكتب – وكيل Office-Agent. أما فى حالة فئة طبيب – وكيل Physician-Agent فإن الاحتفاظ يكون اختياراً . وبالنسبة للاحتفاظ الإجبارى فيكون ممثلاً فى حالة فئة وكيل–عميل Agent-Client.

تبحر قاعدة البيانات Database Navigation :

كما وضح من قبل فإن فئات النموذج التشاورى للغة نظام البيانات CODASYL تشكل الوصلات بين الملفات. إن برنامج قاعدة البيانات يمر خلال سجلات الملف مستخدماً الوصلات Links المزودة بواسطة فئات النموذج التشاورى للغة نظام البيانات CODASYL للوصول إلى ملف معين أو يستمر فى ذلك إلى أن يتم الحصول على النتائج المطلوبة.

وهذه الحركة خلال سجلات الملف مروراً بالوصلات (فئات النموذج التشاورى للغة نظام البيانات CODASYL) تسمى تبحر قاعدة البيانات. حيث يتم معالجة البيانات عادة من المخططات الفرعية (المنظورات الخارجية). وتركز معالجة البيانات فى الحصول على السجل المناسب وعمل التعديلات والتحديثات المطلوبة عليه . ولكى لا نسهب فى تفاصيل كثيرة فإن التبحر داخل قاعدة البيانات يبنى على جملة "أوجد" Find Statement. وتعنى جملة "أوجد" البحث عن سجل معين فى قاعدة البيانات وجعله سجلاً أكثر تداولاً حديثًا . وتبنى جملة المعالجة على أوامر الإضافة Insert والحذف Delete والتعديل modify.

سمات قواعد البيانات الشبكية :

١- لا يوجد حصر عن كيفية تجمع السجلات داخل الوصلات.

٢- سجل الأصل root ليس ابنًا فى أى وصلة.

سلامة البيانات بالنموذج الشبكى :

١- الالتزام بقاعدة النموذج الهرمى التى تقضى بعدم أضافة سجل ابن لو لم يكن سجل الأب موجوداً.

٢- ليس من الضرورى معرفة المفتاح الأساسى أو المفاتيح الخارجية.

مثال ذلك : يوجد نوعان من علاقات الربط relationships بالإضافة إلى واحد - لمتعدد فى علم قواعد البيانات الشائع :

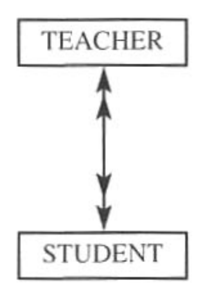
- قواعد بيانات شبكية بسيطة .
- قواعد بيانات شبكية معقدة .

ففى سياق قواعد البيانات الشبكية البسيطة لا يوجد أكثر من رسم محدد يتكون من العديد من علاقات الربط واحد-لمتعدد . أما فى قواعد البيانات الشبكية المعقدة لا بد من تتبع الرسم الموجه . وكل من الشبكات المعقدة والبسيطة يجب أن تفكك إلى العديد من العلاقات واحد - لمتعدد.

فعلى سبيل المثال الذى يوضح العلاقة متعدد - لمتعدد فى قواعد البيانات الشبكية مثال المدرس والطلاب كما هو مبين فى الشكل رقم (٣-١٥). حيث تمثل علاقة الربط متعدد - لمتعدد المدرس Teacher والطلاب Students حالة مهمة تختلف عن النموذج الهرمى . فكل ملف "مدرس" TEACHER يحتفظ بحقل رقم المدرس (#SS) وحيداً. وحقل أسم المدرس TName، وحقل الدرجة الوظيفية Position وحقل الإدارة التابع لها Dept. كل ملف "الطلاب" STUDENTS يحتفظ بحقل رقم الطالب (SSNo) وحقل اسم الطالب Sname وحقل العنوان Address وحقل رقم التليفون (Tel No) وحقل الفصل Classification. حيث إن علاقة الربط بين المدرس والطالب هى متعدد - لمتعدد فإن كل سجل طالب يجب أن يتكرر لكل سجل مدرس حيث المدرس يدرس للطالب.

شكل رقم (٣-١٥) الرسم التخطيطى لهيكل البيانات الشبكية لتمثيل

علاقة الربط متعدد - لمتعدد



معالم نماذج البيانات التبحرية Featare of Navigational Data model :

فى العشرين عاماً الماضية تم استثمار وقت ومجهود كبيرين فى:

١- تصميم قواعد البيانات التبحرية لصعوبة تنفيذ التطبيقات التى تستعملها.

٢- نظم قواعد البيانات التبحرية حققت قدراً كبيراً من القواعد فى تقليل تكرار البيانات وتحسين سلامتها.

٣- كذلك أمكن التغلب على مشكلة لغة الاستعلام التى تتضمن مستويين فى النماذج التبحرية بتجهيز دوارة Loop لتضع مجموعة السجلات فى إطار لغة برمجة ذات مستوى عالٍ high-level.

محدودية نماذج البيانات التبحرية Limitation of the navigational Data Model :

١- لغة الاستعلام المرتبطة بالنماذج الهرمية أو الشبكية تتكون من مجموعة من الأوامر كل منها يتعامل مع سجل واحد فقط كل مرة . وتسمى لغة الاستعلام ذات المستوى المنخفض Low - Level [٣].

٢- تزيد نماذج البيانات التبحرية من متاعب البرمجة؛ لأن مخطط البرامج يقوم بجعل النموذج التبحرى مثالياً بشكل يدوى، فى حين أن نموذج البيانات العلاقى يمتلك نظامه إمكانية جعل الاستعلام نموذجياً (مثالياً) ؛ لأنه يعتمد على أساس الجبر العلاقى[٤].

٣- اعتماد النماذج الهرمية والشبكية على أساس المؤشر Pointer ، سبب متاعب كثيرة في أن جعل هذه النماذج غير مرنة . بل وأكثر من ذلك أن النموذج الشبكي شكل صعوبة بالغة في تغيير أى علاقة بمجرد أن يتم تصميمه[٥].

٤- يتم إعادة التصميم مرة أخرى عند إضافة أو حذف أى حقل.

٥- يتطلب مساحة واسعة لتخزين البيانات عما يتطلبه النموذج العلاقي لنفس الحجم من البيانات[٦] .

٦- بطيء الاستجابة نسبياً نتيجة التبحر إذا ما تم مقارنته بالنموذج العلاقى.

الهوامش :

1- [DATE,1990], C.J. Date, **An Introduction to Database Systems**, Volume I, Fifth Edition, Addison-Wesley publishing Company Inc.,1990.

2- [GRANT, 1987], John Grant, **Logical Introduction to Databases**, Harcourt Brace Jovanovich, Publishers and its subsidiary, Academic Press, 1987.

3- [NAPS, 1986], Thomas L. Naps, and Bhagat Singh, **Introduction to Data Structures with Pascal**, West Publishing Company, 1986.

4- [GILLENSON, 1987], Mark L. Gillenson, 'The Duality of Database Structures and Design Techniques', **Communications of the ACM**, Vol. 30, Number12; December 1987.

5- [GRHAM, 1991], Ian Grham, BIS Applied Systems,**Object-Oriented Models**, Addison-Wesley Publishing Company Inc., 1991.

6- [GILLENSON, 1990], Mark L. Gillenson, 'Physical Design Equivalencies in Database Conversion', **Communications of the ACM**, Vol.33, Number 8. August 1990.

7- [Connolly 1996], Thomas M. Connolly and Carolyn E. Begg, **Database Systems**, Addison-Wesley Publishing Co., Inc., 1996.

الفصل الرابع
نموذج البيانات العلاقى

مقدمة :

لقد حصل نموذج البيانات العلاقى حديثًا على انتشار فإن جميع نماذج البيانات الأخرى. ويرجع السبب وراء ذلك إلى قلة تكاليف تصميم التطبيقات الخاصة به وسهولتها، بالإضافة إلى ارتكازه على الأساس العلمى لنظرية الفئات الرياضية؛ لذا كان من الضرورى استعراض الموضوعات التى سيتم تقديمها فى هذا الفصل.

تعريف النموذج العلاقى :

يتم تمثيل البيانات داخل هذا النموذج من خلال جدول (ملف) ذى بعدين ، يسمى الجدول العلاقى (العلاقة Relation). وتتكون قاعدة البيانات العلاقية من مجموعة جداول علاقية يتم الترابط فيما بينها باستخدام علاقات الربط Relationships.

مكونات النموذج العلاقى :

يتكون النموذج العلاقى من المكونات التالية:

- هيكل البيانات.
- عوامل معالجة البيانات.
- قواعد سلامة قاعدة البيانات العلاقية.

ويتكون هيكل البيانات من:

- مخططات الجداول العلاقية التى تتكون بدورها من مجموعة خصائص.
- والوقائع التى تمثل القيم الملحقة بتلك المخططات.

وتستخدم نظم قواعد البيانات العلاقية عوامل معالجة البيانات التالية:

- الجبر العلاقى Relational Algebra
- الحساب العلاقى Relational Calculus

ويتم إدراج قواعد سلامة النموذج العلاقى تحت النقاط التالية :

- قواعد سلامة الكينونة أو النطاق.
- قواعد السلامة داخل الجدول العلاقى.
- قواعد السلامة المرجعية.

التبعيات الوظيفية :

تعد التبعيات الوظيفية بمثابة ارتباط خاص بين خاصيتين أو أكثر فى الجدول العلاقى. إحدى هذه الخصائص تكون تابعاً وظيفياً للخاصية الأخرى.

كما سيتم التطرق إلى بديهيات التبعيات الوظيفية وقواعد استنتاجها وكيفية حساب الفئة الانغلاقية.

سمات ومحدوديات النماذج العلاقية :

سيتم التطرق بشكل مختصر عن سمات الجداول العلاقية ومحدودية هذه النماذج.

تعريف نموذج البيانات العلاقى :

تمثل البيانات فى نموذج البيانات العلاقى بجدول (ملف) ذى بعدين يسمى الجدول العلاقى (العلاقة relation) . وكل عمود (حقل) Column فى الجدول العلاقى هو خاصية attribute. فى حين أن صفوف (سجلات) rows الجدول العلاقى هى مجموعات من القيم المرتبة tuples ، وكل قيمة مرتبة تمثل كينونة. وبيانات كل خاصية فى الجدول العلاقى تتكون من نطاق من القيم domain. وتمثل نطاقات القيم بالجداول العلاقية سقف قاعدة البيانات. وكل الجدول العلاقى له درجة Arity تحدد بعدد الخصائص ، فى حين يشكل عدد مجموعات القيم المرتبة (عدد الكينونات) تعددية Cardinality الجدول العلاقى[١].

ويتم إنشاء علاقات الربط relationships بين جداول قاعدة البيانات العلاقية باستخدام المفتاح الأساسى Primary Key الممثل فى النموذج العلاقى ، وقيمة هذا المفتاح لا تتكرر فى مجموعات القيم المرتبة Tuples ، وعليها يتم تعريف مجموعات القيم المرتبة (الكينونات). ويبين الشكل رقم (٤-١) نموذج البيانات المنطقى Logical Data Model لعلاقة الربط بين الجدول العلاقى الخاص "بالإدارة" DEPARTMENT والجدول العلاقى الخاص "بالمدير" MANAGER والتى تمثل بعلاقة ربط واحد - لواحد.

شكل رقم (٤-١) نموذج البيانات المنطقى لعلاقة ربط بين الجدولين العلاقيين "الإدارة" و"المدير"

DEPARTMENT	—	MANAGER

مكونات النموذج العلاقى :

يتكون النموذج العلاقى من :

١- هيكل بيانات.

٢- عوامل معالجة البيانات.

٣- قواعد سلامة قاعدة البيانات العلاقية.

أولاً - هيكل البيانات Data Structure :

مخطط الجدول العلاقى والوقائع :

تنظم البيانات فى النظم العلاقية على المستويين الخارجى والمفاهيمى ، كما سبق ووضح التعريف السابق ، فى صورة جداول علاقية والتى تعرف بالعلاقات relations ، وكل علاقة تعبر عن نوع كينونة entity type.

ويمكن تمثيل نموذج البيانات المنطقى بين الجدولين العلاقيين "الإدارة" و"المدير" فى نموذج علاقى يتكون من مخططى الجدولين العلاقيين. أحدهما يمثل "الإدارة" DEPARTMENT والآخر يمثل المدير MANAGER. ويبين الشكل رقم (٤-٢) المخطط الخاص بكل من الجدولين العلاقيين. ويجب التفرقة بين مخطط الجدول العلاقى والواقعة. حيث يمثل المخطط فى الجدول العلاقى أسماء الخصائص المكونة له فقط، فى حين تمثل الواقعة القيم (البيانات) الملحقة بهذا المخطط.

شكل رقم (٤-٢) المخططات الخاصة بكل من الجدولين العلاقيين "الإدارة" و"المدير"

DEPTMENT (D-ID,DName,DAddr);

MANAGER (D-ID,M-ID, MTitle,MName);

ويمكن إلحاق الوقائع (القيم) الخاصة بالمخططات الخاصة بالجدولين العلاقيين "الإدارة" و "المدير" كما فى الشكل رقم (٤-٣ أ ، ب).

شكل رقم (٤-٣أ) تمثيل القيم (الوقائع) فى الجدول العلاقى "الإدارة"

DEPARTMENT

D - ID	D - NAME	D - ADDR
101	Engineering	101
102	Computer Science	102
103	Biology	103
104	Medical Technology	104

شكل رقم (٤-٣ب) تمثيل القيم (الواقعة) وعلاقات الربط فى الجدول العلاقى "المدير"

MANAGER

M - ID	D - ID	M - TITLE	M - NAME
MG 1	101	Chief Scientist	Mr. Brown
MG 2	102	Systems Designer	Mr. Charles
MG 3	103	Sr. Biologist	Dr. Green
MG 4	104	Sr. Technologist	Mr Care

فى الشكل رقم (٤-٣أ) ، تمثل خاصية المعرف الخاصة بالإدارة D_ID المفتاح الأساسى للجدول العلاقى "الإدارة" DEPARTMENT فى حين أن فى الشكل رقم (٤-٣ب) تمثل خاصية المعرف الخاصة بالمدير M_ID المفتاح الأساسى للجدول العلاقى "المدير" MANAGER فى حين تمثل خاصية المعرف الخاصة بالإدارة D_ID فى ذلك الجدول المفتاح الخارجى. ويمكن أيضًا الحصول على خاصية أو مجموعة خصائص لتعريف مجموعات القيم المرتبة Tuples. هذه الخصائص تسمى المفاتيح المرشحة.

ثانياً : عوامل معالجة البيانات Data Manipulation Operators :

تمكن العوامل العلاقية المستفيدين من إجراء عمليات الفئات Set Operations على جدول علاقى أو أكثر وهى ما تسمى بالجبر العلاقى والعوامل الأكثر

استخداماً هى الاختيار الرأسى PROJECT، الاختيار الأفقى SELECT، الربط JOIN ، والاتحاد UNION.

وتستخدم معظم نظم إدارة قواعد البيانات العلاقية الحساب العلاقى Relational Calculus. ويعتمد الحساب العلاقى على الاسلوب المنطقى وهو مكافىء تماماً للجبر العلاقى كأساس للغات الاستعلام مثل SQL فى لغة DB2 و QUEL فى لغة الإنجرس , SQL*PLUS فى لغة الأوراكل ORACLE. وفيما يلى عرض مبسط للعوامل العلاقية والأسس الفنية بكل من الجبر والحساب العلاقى(٢)،(٣) .

١- **الجبر العلاقى** Relational Algebra :

لازال المحتوى الرئيسى للنموذج العلاقى هو الجبر العلاقى الذى يتكون من مجموعة من العمليات ، كما هو مبين فى الشكل رقم (٤-٤) ، التى يمكن تقسيمها إلى مجموعتين:

(أ) مجموعة العمليات التقليدية :

مثل الاتحاد Union ، التقاطع Intersection ، الطرح Difference والضرب الكارتيزى . Cartrsian Product .

(ب) مجموعة العمليات العلاقية الخاصة :

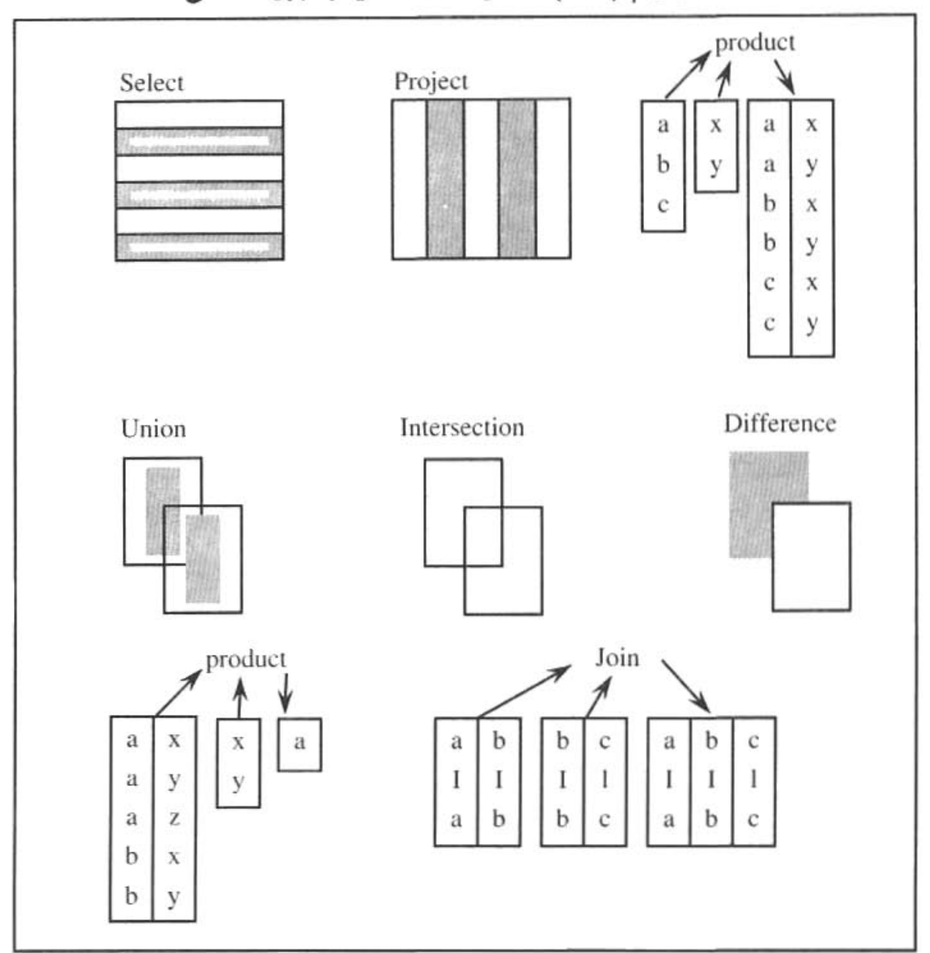
مثل الحصر الأفقى Select ، الاختيار الرأسى Project ، الربط Join ، والقسمة divide.

- تعريف درجة الجدول العلاقى الثنائية جبرياً :

تختلف درجة الجدول العلاقى (العلاقة) عن درجة علاقة الربط. حيث تمثل الأولى عدد الخصائص بالجدول العلاقى فحين تمثل الأخيرة عدد أنوا ع الكينونات المشاركة فى علاقة الربط والتى سبق وتم شرحها فى الجزء السابق. وتتشابة التعددية فى كل من الجدول العلاقى وعلاقة الربط. ويمكن الحصول على الجدول العلاقى R من خلال تحديد مدى الارتباط بين الخاصتين x و y ، واللتان يمكن التعبير عنهما جبرياً كالآتى(٤) :

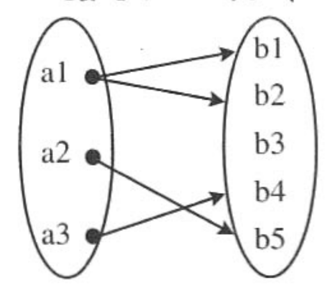
$R : \chi \rightarrow y$

الشكل رقم (٤-٤) العمليات الأساسية بالجبر العلاقى



حيث إن (R) تشير إلى الجدول العلاقى الذى ينشأ بين الخاصتين (x) ، (y). ويشير السهم (⟵) إلى أن اتجاه الرابطة من الخاصية (x) إلى الخاصية (y) كما هو موضح بالشكل رقم (٤-٥).

شكل رقم (٤-٥) الرابطة السهمية بين الخاصتين x ، y



ويمكن التعبيرعنها بالجبر العلاقى كالآتى :

$$R: x \rightarrow y = \{(x , y) \mid x \in Domain (X) \wedge y \in Domain (Y)\}$$

حيث إن الرموز السابق ورودها بالتعبير العلاقى تعبر عن الآتى :

{ } : بداية ونهاية الفئة.

| : حيث إن.

(x , y) : تشير إلى قيم الزوج المرتب داخل الفئة.

$\in$: تنتمى إلى.

(X) : نطاق قيم الخاصية x .

(Y) : نطاق قيم الخاصية y .

$\wedge$: تعنى تقاطع الخاصية (Y) مع الخاصية (X) .

من الرابطة السهمية الممثلة بالشكل رقم (٤-٥) يمكن الحصول على الجدول العلاقى R كالآتى :

R = { (a1,b1) , (a1,b3) , (a2,b5) ,(a3,b4)}

والتى يمكن وضعها فى جدول كما هو موضح بالشكل رقم (٤-٦) كالآتى :

شكل رقم (٤ -٦) يوضح الجدول العلاقى R حسب تعريف الفئات الجبرية

R

x	y
a1	b1
a1	b3
a2	b5
a3	b4

تعريف درجة الجدول العلاقى اللاثنائية جبريًا :

هذه الدرجة تمثل فى حالة درجات العلاقات الثلاثية 3 - ary والرباعية 4 - ary وغيرها كالآتى :

$R(x,y,z) = \{(a, b, c) \mid a \in Domain(X) \wedge b \in Domain(Y) \wedge c \in Domain(Z)\}$

والاختلاف بين درجة العلاقة الثنائية ودرجات العلاقات الأخرى (الثلاثية والرباعية وغيرها) يكمن فى عدم معرفة اتجاه الرابطة لوجود أكثر من نطاق للقيم وإن كان من الصعوبة بمكان تحديد الاتجاه.

من هذه التعريفات الجبرية يمكن تعريف الجدول العلاقى (العلاقة) بأنه يتكون من المخطط Schema والوقائع .Instances والمخطط هو أسماء الخصائص Z,Y,X فى الجدول العلاقى R كما يلى :

R (X,Y,Z)

والوقائع تختلف فى التفسير عن المخطط حيث إنها تكافىء القيم التى تحل تحت كل اسم خاصية، وتشكل نطاقات القيم المختلفة لكل الخصائص. ومن هذا التوضيح يتبين أن مخطط العلاقة يظل ثابتًا فى حين أن وقائع العلاقة تتغير بسبب تغير مجموعات القيم المرتبة tuples وحذفها وتحديثها. ومن ثم يتبين أن الجداول العلاقية تخضع لقواعد نظرية الفئات والتى يمكن استخدام عمليات الخاصة بها على هذه الجداول العلاقية.

العمليات الخاصة بنظرية الفئات :

قبل البدء فى الدخول إلى تفاصيل العمليات الخاصة بنظرية الفئات فى الجبر العلاقى ينبغى التطرق للمثال رقم (٤-١) [٣] ، [٤] ، [٢].

مثال (٤-١) :

بفرض وجود الجدولين العلاقيين A و B التاليين ذَوَى الدرجة الرباعية والتعددية الثنائية لكل منهما كما هو مبين بالشكل رقم (٤-٧).

الشكل رقم (٤-٧) يبين الجدولين العلاقيين A و B

A

S #	Sname	Status	City
S 1	Samy	20	Jeddah
S 4	S #	20	Jeddah

B

S #	Sname	Status	City
S 1	Samy	20	Jeddah
S 2	Jamal	10	Dammam

مجموعة العمليات التقليدية :

١. الاتحاد Union :

يمثل اتحاد الجدولين العلاقيين (العلاقتين) A و B مجموعة القيم المرتبة tuples فى الجدول العلاقى A أو الجدول العلاقى B أو كليهما ويرمز له جبريًا بـ :

$A \cup B$

ولكن يشترط أن تكون درجة الجدول العلاقى arity فى كلا الجدولين العلاقيين A و B متساوية. بمعنى آخر فإنَّ مخطط كلا الجدولين العلاقيين A و B ينبغى أن

يتساوى هيكليًا ووقائعهم تتساوى دلاليًا ، أى تستنتج من نفس نطاق القيم لكلا الجدولين العلاقيين ومن ثم لو فرض أن:

(n) تمثل عدد مجموعات القيم المرتبة tuples فى الجدول العلاقى A .

(m) تمثل عدد مجموعات القيم المرتبة tuples فى الجدول العلاقى B .

فإن تعددية Cardinality مجموعات القيم المرتبة فى الجدول العلاقى الجديد N1 تكون أقل من (n + m) أو تساويها ، بينما تظل درجة الجدول العلاقى arity الناتج ثابتة كما هو موضح بالشكل رقم (٤-٨) عن المثال رقم (٤-١).

الشكل رقم (٤-٨) والجدول العلاقى N1 الناتج عن اتحاد الجدولين العلاقيين A و B

N1

S #	Sname	Status	City
S 1	Samy	20	Jeddah
S 4	Ahmed	20	Jeddah
S 2	Jamal	10	Dammam

٢- الطرح Difference :

يمثل الجدول العلاقى الناتج عن طرح الجدول العلاقى A من الجدول العلاقى B بمجموعات من القيم المرتبة tuples ويرمز لها جبريًا بـ :

A - B

ولكن يشترط أن يكون درجة الجدول العلاقى فى كلا الجدولين العلاقيين متساوية، وأن يتطابق نطاقات قيم الخصائص المتناظرة ، ومن ثم لو فرض أن:

(n) عدد مجموعات القيم المرتبة tuples فى الجدول العلاقى A .

(m) عدد مجموعات القيم المرتبة tuples فى الجدول العلاقى B.

فإن تعددية مجموعات القيم المرتبة Cardinality بالجدول العلاقى الجديد N2 تكون أقل من (n)؛ لأنها تمثل مجموعات القيم المرتبة التى تنتمى إلى الجدول العلاقى،

A ولا تنتمى إلى الجدول العلاقى B، ومع ذلك تظل درجة الجدول العلاقى الناتج arity ثابتة ، كما هو مبين بالشكل رقم (٤-٩) عن المثال (٤-١).

الشكل رقم (٤-٩) والجدول العلاقى N2 الناتج عن طرح الجدول العلاقى A من الجدول العلاقى B

N2

S #	Sname	Status	City
S 4	Ahmed	20	Jeddah

٣- التقاطع Intersection :

يمثل الجدول العلاقى الناتج عن تقاطع الجدول العلاقى A مع الجدول العلاقى B بمجموعات القيم المرتبة tuples التى تنتمى إلى كل من الجدول العلاقى A والجدول العلاقى B ويرمز لها جبرياً:

A B = A - (A - B)

ويلاحظ من التعبير الجبرى أن التقاطع بين الجدولين العلاقيين A و B يمكن الحصول عليه باستخدام عمليات الطرح كما هو مبين. ويشترط أن تكون درجة الجدول العلاقى فى كلا الجدولين العلاقيين متساوية، وأن يتطابق نطاقات القيم المتناظرة فى كلا الجدولين العلاقيين. ومن ثم لو فرض أن :

(n) عدد مجموعات القيم المرتبة فى الجدول العلاقى A.

(m) عدد مجموعات القيم المرتبة فى الجدول العلاقى B.

فإن تعددية مجموعات القيم المرتبة Cardinality فى الجدول العلاقى الناتج N3 تكون أصغر من (n) أو (m). وتظل درجة الجدول العلاقى arity ثابتة كما هو مبين بالشكل رقم (٤-١٠) عن المثال (٤-١).

الشكل رقم (٤-١٠) والجدول العلاقى N3 الناتج عن تقاطع الجدولين العلاقيين A و B

N3

S #	Sname	Status	City
S 1	Samy	20	Jeddah

٤- الضرب Product :

يمثل الجدول العلاقى الناتج من حاصل الضرب الكارتيزى للجدول العلاقى A فى الجدول العلاقى B بمجموعات القيم المرتبة. وتكون درجة الجدول العلاقى الناتج arity تساوى درجة الجدول العلاقى A مضافًا إليها درجة الجدول العلاقى B ويرمز لها جبريًا بـ :

A * B

ومن ثم لو فرض أن :

(n) عدد مجموعات القيم المرتبة فى الجدول العلاقى A .

(Thirteen) عدد مجموعات القيم المرتبة فى الجدول العلاقى B .

فإن تعددية مجموعات القيم المرتبة Cardinality فى الجدول العلاقى الناتج تساوى حاصل ضرب m*n. وتساوى درجة الجدول العلاقى الناتج arity: درجة الجدول العلاقى A مضافًا إليها درجة الجدول العلاقى B.

وفى حاصل الضرب الكارتيزى يكون الجدول العلاقى الناتج متضمنًا مخطط الجدول العلاقى A ملاصقًا Cancatinate لمخطط الجدول العلاقى B. ويمكن توضيح ذلك بطريقة مبسطة من خلال المثال (٤-٢).

مثال (٤-٢) :

بافتراض وجود الجدولين العلاقيين X و Y التاليين ذوى الدرجة الأحادية، والتعددية الثلاثية للجدول العلاقى X والثنائية للجدول العلاقى Y كما هو مبين بالشكل رقم (٤-١١).

الشكل رقم (٤-١١) يبين الجدولين العلاقيين X و Y

X

S #
S 1
S 2
S 3

Y

P #
S 1
S 2

ويكون حاصل الضرب الكارتيزى للجدولين علاقيين X و Y الجدول العلاقى N4 كما هو موضح بالشكل رقم (٤-١٢).

الشكل رقم (٤-١٢) حاصل الضرب الكارتيزى للجدولين العلاقيين X و Y

N4

S #	P #
S1	P1
S1	P2
S2	P1
S2	P2
S3	P1
S3	P3

مجموعة العمليات العلاقية الخاصة :

١- الحصر الأفقى Selection :

يمثل ناتج الحصر الأفقى للجدول العلاقى R مجموعات القيم المرتبة tuples التى تحقق الشرط المطلوب ، والذى يمكن أن يرمز له بالرمز (Θ) ويرمز لعملية الحصر الأفقى بالرمز (σ) وسواء كان هذا الشرط معقداً أم بسيطاً لكن ناتجه سيكون صحيحاً أو خطأ ويعبر عنه جبرياً كالآتى:

$$\sigma_{\Theta}(R)$$

مثال (٤-٣) :

بافتراض وجود الجدول العلاقى R كما هو مبين بالشكل رقم (٤-١٣)، فيمكن توضيح الاختيار الأفقى والذى يشير إلى مجموعات القيم المرتبة tuples فى الجدول العلاقى R الذى يحقق شرط الحصر التالى:

$$\sigma_{2>3}(R)$$

ويتحقق هذا الشرط عندما تكون قيمة الخاصية attribute الثانية بكل قيمة مرتبة أكبر من قيمة الخاصية الثالثة بنفس القيمة المرتبة. ويكون الجدول العلاقى الناتج له

نفس درجة arrity الجدول العلاقى R ، وأن تعددية Cardinality مجموعات القيم المرتبة لذلك الجدول العلاقى تكون أقل من تعددية مجموعات القيم المرتبة فى اا الجدول العلاقى أو تساويها .

الشكل رقم (٤-١٣) الجدول العلاقى R

R

R #	Value 1	Value 2	Location
R 1	30	20	London
R 2	15	27	Paris
R 3	40	18	Athena
R 4	55	67	Cairo

والجدول العلاقى N5 الناتج والمحقق للشرط السابق يمكن الحصول عليه كما هو موضح فى الشكل رقم (٤-١٤).

الشكل رقم (٤-١٤) الجدول العلاقى الناتج تحقيقًا لعملية الحصر الأفقى (٤-٣).

N 5

R #	Value 1	Value 2	Location
R 1	30	20	London
R 3	40	18	Athena

ويتضمن التعبير الشرطى Θ ما يلى :

١- قيمتا الشرط Operands ، فإما أن تكون قيمًا ثابتة وإما أن تحدد رقم الخاصية

٢- عوامل المقارنة العلاقية هى :

$> , < , \leq , \geq , \equiv , \neq$

عوامل المقارنة المنطقية هى :

$\neg , \wedge , \vee$

٢- الاختيار الرأسى Projection :

هو عملية أحادية تتم على الجدول العلاقى وتحذف بعض الخصائص وتعيد ترتيب الخصائص المتبقية. ولو فرض أن الجدول العلاقى الذى تتم عليه عملية الاختيار الرأسى هو الجدول العلاقى R الموضح بالمثال رقم (٤-٣). ويرمز لعملية الاختيار الرأسى بالرمز (∏). وعلى سبيل المثال يمكن التعبير عن عملية اختيار خاصية رقم المعرف #R وخاصية المكان Location جبرياً كالآتى :

$$\Pi_{(S\#,Location)}(R)$$

يحتوى الجدول العلاقى الناتج عن عملية الاختيار الرأسى على كل من الخاصيتين رقم المعرف #R والمكان Location. أما بقية الخصائص المضمنة داخل الجدول العلاقى R فإنها تختفى. ويتضح أن درجة الجدول العلاقى arity الناتجة أقل من درجة الجدول العلاقى R أو تساويها وكذلك تعددية مجموعات القيم المرتبة كما هو موضح بالشكل رقم (٤-١٥).

الشكل رقم (٤-١٥) الجدول العلاقى R بعد عملية الاختيار الرأسى

R

R #	*Location*
R1	London
R2	Paris
R3	Athena
R4	Cairo

٣- الربط Join :

ويتم تنفيذ عملية الربط الطبيعى Natural Join وحسابها طبقاً للعمليات التالية :

١- الضرب الكارتيزى للجدولين العلاقيين.

٢- يتم الحصر الأفقى من حاصل الضرب الكارتيزى ذى القيم المتوافقة.

٣- لخصائص الجدول العلاقى الأول مع خصائص الجدول العلاقى الثانى.

٤- يتم عمل اختيار رأس للخصائص طبقًا للشروط المحددة.

ومن ثم لو فرض أن :

(Fourteen) عدد مجموعات القيم المرتبة فى الجدول العلاقى الأول.

(m) عدد مجموعات القيم المرتبة فى الجدول العلاقى الثانى.

فان تعددية Cardinality مجموعات القيم المرتبة فى الجدول العلاقى الناتج تعادل عدد مجموعات القيم المرتبة للجدول العلاقى الأول بالإضافة إلى عدد مجموعات القيم المرتبة الجدول العلاقى الثانى. أما درجة الجدول العلاقى الناتج فتكون أقل من الخصائص التى عددها (m + n) لكلا الجدولين العلاقيين معًا أو تساويها ويتم توضيح ذلك فى المثال رقم (٤-٤).

مثال (٤-٤) :

بافتراض وجود وجود الجدولين العلاقيين S و P كما بالشكل رقم (٤-١٦).

S

S #	Sname	SStat	City
S 1	Samy	20	Jeddah
S 2	Ali	15	Ryiadh
S 3	Ahmed	20	Jeddah

P

P #	Pname	PStat	City
P 1	Samy	20	Jeddah
P 2	Jamal	10	Dammam
P 3	Hany	17	Ryiadh
P 4	Osama	12	Ryiadh

وتكون نتيجة الربط الطبيعى للجدولين العلاقيين P و S هو الجدول العلاقى N6 حسب خاصية المدينة City كما هو موضح بالشكل رقم (٤-١٦).

الشكل رقم (٤-١٦) نتيجة الربط الطبيعى للجدولين العلاقيين P و S حسب خاصية المدينة City

N6

S #	Sname	SStat	City	P #	Pname	PStat
S 1	Samy	20	Jeddah	P 1	Samy	20
S 2	Ali	15	Ryiadh	P 3	Hany	17
S 2	Ali	15	Ryiadh	P 4	Osama	12
S 3	Ahmed	20	Jeddah	P 1	Samy	20

٤- القسمة Division :

وفيها يتم الحصول على مجموعات القيم المرتبة فى الجدول العلاقى (المقسوم) المتكررة مع كل عنصر من العناصر التى تنتمى إلى الجدول العلاقى (المقسوم عليه) كما هو موضح بالمثال رقم (٤-٥).

مثال (٤-٥) :

بافتراض وجود الجدول العلاقى (المقسوم) D1 والجدول العلاقى (المقسوم عليه) E1 كما هو موضح بالشكل رقم (٤-١٧).

الشكل رقم (٤-١٧) يبين الجدول العلاقى (المقسوم) D1 والجدول العلاقى (المقسوم عليه) E1

D1

A	B	C	D
a	b	c	d
a	B	e	f
b	C	e	f
e	D	c	d
e	D	d	e
e	D	e	f

E1

C	D
c	d
e	f

الشكل رقم (٤-١٨) يبين الجدول العلاقى ناتج القسمة

D1 / E1

A	B
a	b
e	d

٢- الحساب العلاقى Relational Calculus :

إنه من الممكن إثبات أن أى مجموعة وظائف للجداول العلاقية relations يمكن التعبير عنها فى الجبر العلاقى ، ويمكن أيضًا التعبير عنها باستخدام الحساب العلاقى. وتنقسم العمليات بالحساب العلاقى إلى مجموعتين [٧،٤،٢] :

(أ) الحساب العلاقى الخاص بمجموعات القيم المرتبة tuples .

(ب) الحساب العلاقى الخاص بنطاق القيم domain .

(أ) الحساب العلاقى الخاص بمجموعات القيم المرتبة :

يمكن تعريف الجدول العلاقى فى حدود مجموعات القيم المرتبة التى تستعمل فى تمثيل الاستفسار تحت بعض الشروط. هذه الشروط يمكن صياغتها باستخدام مجموعة القيم ظ الفردية atoms المرتبطة بالعمليات المنطقية. هذه القيم الفردية العلاقية يمكن ان تكون متغيرات مجموعات القيم المرتبة (tuples variables (tv. والمتغير هو اسم يمكن استبداله بقيمة. ومن الرموز المستخدمة : السهم ذو الاتجاه الواحد (→) وهو يعنى "إذا ثم" والسهم ذو الاتجاهين (↔) وهو يعنى "إذا وإذا فقط".

خواص متغيرات مجموعات القيم المرتبة (tv)

(١) صلة المتغير بالجدول العلاقى :

بافتراض أن A ترمز إلى اسم الجدول العلاقى، وأن t ترمز إلى متغير يعبر مجموعة قيم مرتبة. عندئذ تكون A (t) صحيحة لو أن متغير مجموعة قيم مرتبة (t) هو مجموعة القيم المرتبة فى الجدول العلاقى A وتكون خطأ بخلاف ذلك. بمعنى أن (t) لا تنتمى إلى A . مثال ذلك فى الجدول العلاقى A فى الشكل رقم (٤-٧) نجد أن :

$$t = (S\ 1, \text{Samy}\ , 20\ , \text{Jeddah}) \in A$$

(٢) صلة متغير بمتغير آخر :

يمكن التعبير عن صلة متغير بمتغير آخر كالآتى :

$t = [1]\ \chi\ ^{\chi}\ [3]$

وذلك بافتراض أن الرمز (χ) يشير إلى أن عوامل الربط العلاقية وأن الرمزين (t) و (x) يشيران إلى مجموعة قيم مرتبة بالجدول العلاقى محل الاستعلام، وأن الأرقام [١] و [٣] تشير إلى أرقام مجموعات القيم المرتبة بالجدول العلاقى. ومن ثم فى التعبير السابق لو كان عامل الربط العلاقى (_) يكافئ (=) فإن الشرط يصبح صحيحًا فى حالة تطابق مجموعة القيم المرتبة رقم (١) مع مجموعة القيم المرتبة رقم (٣).

(٣) صلة المتغير بثابت عددى :

تحدث صلة المتغير بثابت عددى عندما يتم مقارنة متغير لمجموعة قيم مرتبة بقيمة ثابتة. وإنه من الممكن ربط أى ثلاث خواص معًا باستخدام عوامل الربط المنطقية لتكوين الصيغة المنطقية . ويمكن التعبير عن ذلك بـ :

$t\ [1]\ \chi\ \text{Const}$

الصيغ التعبيرية باستخدام الحساب العلاقى :

مثال (٤-٦) :

لتوضيح كيفية تطبيق واستخدام الصيغ التعبيرية فى الحساب العلاقى الخاص بكل من مجموعات القيم المرتبة ونطاق القيم ، يمكن استخدام قاعدة بيانات (افتراضية) مكونة من ثلاثة جداول علاقية المبينة بالشكل رقم (٤-١٩) هى:

- الجدول العلاقى الخاص "بالموردين" SUPPLIERS ويرمز له بالرمز S.
- الجدول العلاقى الخاص "بقطع الغيار" PARTS ويرمز له بالرمز P.
- الجدول العلاقى الخاص بعلاقة الربط بين كل من جدولى "الموردين" و "قطع الغيار" ويرمز له بالرمز SP.

الشكل رقم (٤-١٩) قاعدة بيانات علاقية (افتراضية) خاصة بموردى قطع الغيار

S

S #	Sname	Stat	City
S 1	Samy	20	Jeddah
S 2	Jamal	10	Dammam
S 3	Magdy	30	Dammam
S 4	Ahmed	20	Jeddah
S 5	Assem	30	Ryiadh

S

P #	Pname	Color	Weight	City
P 1	Nut	Red	12	Jeddah
P 2	Bolt	Green	17	Dammam
P 3	Screw	Blue	17	Khobar
P 4	Screw	Red	14	Jeddah
P 5	Cam	Blue	12	Dammam
P 6	Cog	Red	19	Jeddah

SP

S#	P#	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400
S5	P1	600
S5	P3	400

وفيما يلى أمثلة الصيغ التعبيرية فى الحساب العلاقى الخاص بكل من مجموعات القيم المرتبة ونطاق القيم :

(١) إيجاد أسماء كل الموردين Suppliers (S) الذين يعيشون فى مدينة City جدة : Jeddah

$\{ t(1) \mid S(\chi) \wedge \chi[4] = \text{'Geddah'} \wedge t[1] = \chi[2] \}$

التفسير الحقيقى لرموز الصيغة التعبيرية السابقة :

t (1) : تعنى أن درجة الجدول العلاقى الناتج أحادية (١) لأنها ستمثل أسماء الموردين فقط.

S(x) : تعنى أن متغير مجموعة القيم المرتبة المستخدم هو x بالنسبة للجدول العلاقى S.

x[4] : تشير إلى قيمة الخاصية الرابعة لمتغير مجموعة القيم المرتبة x وهى المدينة (City).

x[2] : تشير إلى قيمة الخاصية الثانية لمتغير مجموعة القيم المرتبة x وهى الاسم (Sname).

الشرط : 'x [4] = Jeddah' وهو يعنى البحث فى قيمة الخاصية الرابعة لمتغير مجموعة القيم المرتبة x عن مدينة جدة Jeddah.

الناتج : t [1] = x [2] تعنى كتابة اسم المورد Sname لمجموعة القيم المرتبة t فى الجدول العلاقى ذات الدرجة الاحادية (١) كما هو موضح فى الشكل رقم (٤-٢٠).

الشكل رقم (٤-٢٠) أسماء الموردين قاطنى مدينة جدة.

Sname
Samy
Ahmed

(٢) إيجاد أسماء الموردين الذين يوردون قطعة الغيار رقم P2 :

$$\{t(1) \mid S(\chi) \wedge SP(z) \wedge \chi[1] = z[1] \wedge z[2] = 'P2' \wedge t[1] = \chi[2]\}$$

الشكل رقم (٤-٢١) أسماء موردى قطعة الغيار P2.

Sname
Samy
Jamal
Magdy
Ahmed

(٣) إيجاد كل أسماء الموردين الذين يوردون قطعة الغيار "Nut":

$$\{t(1) \mid S(\chi) \wedge P(y) \wedge SP(z) \wedge \chi[1] = z[1] \wedge y[1] = z[2] \wedge y[2]='Nu' \wedge t[1] = \chi[2]\}$$

الشكل رقم (٤-٢٢) أسماء موردى قطعة الغيار "Nut".

Sname
Samy
Jamal
Assem

ونلاحظ هنا أنه ليس هناك حاجة إلى تخزين الجدول العلاقى الناتج عن تنفيذ عمليات مجموعات القيم المرتبة ، كما هو ، ففى الاختيار الرأسى والحصر الأفقى وغيرها من العمليات الخاصة بالجبر العلاقى.

وفى الحساب العلاقى الخاص بمجموعات القيم المرتبة tuples ، يتم مراجعة كل مجموعات القيم المرتبة فى الجداول العلاقية relations طبقًا للشروط المفروضة. ولو كانت الشروط مطابقة يتم اختيار الصف بخلاف ذلك تكون النتيجة خطأ. ولكن ماذا يتم فى الحساب العلاقى الخاص بمجموعات القيم المرتبة للناتج الذى لم يتم حفظه ؟. هذا يعنى أن وقت التشغيل run-time سوف ينخفض. ومع ذلك فكل من الجبر العلاقى والحساب العلاقى لمجموعات القيم المرتب متكافئان فى التنفيذ. والجبر العلاقى أكثر سهولة فى صياغة تعبيراته ولكن يحتاج إلى ذاكرة فى الحاسب أكبر نسبيًا من الذاكرة التى يحتاجها الحساب العلاقى الخاص بمجموعات القيم المرتبة.

- محددات قياس المتغيرات Quantifiers :

إنه من الممكن استخدام محددات القياس المتوافرة مثل ($\forall$) التى تعنى (كل) all ومثل ($\exists$) التى تعنى يوجد (على الأقل واحد). ومثال ذلك : ($\forall$ t) يعنى أن كل القيم للمتغير (t) يجب أن تكون صحيحة لكى يكون الناتج صحيحًا. ($\exists$ t) تعنى أن قيمة واحدة تكفى أن تكون صحيحة لكى يكون الناتج صحيحاً.

وليس صحيحًا استخدام هذين التعبيرين (S ∀) أو (S ∃) حيث إن S هذا تعنى جدولاً علاقيًا وليس متغيرًا.

- **المتغير الحـــر Free variable** :

إنه متغير غير مرتبط بأى نوع من أنواع محددات القياس.

- **المتغير المرتبط** Bound variable :

انه المتغير المرتبط بأحد محددات القياس مثل كل (أو) يوجد .

- **التعبير الحر** Free expression :

فى حالة وجود متغير واحد حر فى التعبير ، فإن كل التعبير يُعدُّ تعبيرًا حرًا.

- **التعبير المرتبط** Bound expression :

إنه التعبير الذى تكون به كل المتغيرات مرتبطة. وفى حالة رفض أى قيمة فردية atom فإن الصيغة تكون صحيحة. وهذه الصيغة غير مسموح بها فى الجبر العلاقى.

- **فرضية الغلق Closed word assumption** :

فى حالة رفض متغير ، فإن الرفض يمكن أن يكون أى شىء لو لم يتم تعريف نطاق القيم. مثل هذا النطاق يطلق عليه حيز الفئة التى يتم تمثيلها باستخدام الضرب الكارتيزى لكل نطاقات القيم.

خطوات تمثيل المتغير فى الحساب العلاقى :

يتم تحديد مجموعات القيم المرتبة للجدول العلاقى فى الحساب العلاقى باستعمال طريقة تلقائية. تلك الطريقة تؤدى إلى حل صحيح ولكن ليس بالحل البسيط. ويتمثل هذا التحديد فى الخطوات التالية:

١- تحديد الجدول العلاقى.

٢- لكل جدول علاقى : يتم تعريف المتغير كمجموعة قيم مرتبة غير محددة (كصف ما) فى الجدول العلاقى، وهكذا بالنسبة لكل الجداول العلاقية يتم تعريف مجموعات القيم المرتبة المميزة مثل s, t, r وهكذا.

٣- يأخذ فى الحسبان أن الناتج الذى تمَّ الحصول عليه هو أيضاً جدول علاقى. هكذا تسمى الجداول العلاقية باسماء المتغيرات المميزة لكل الجدول العلاقية التى تم الحصول عليها.

٤- يتم التعبير عن أى من الشروط الإضافية المذكورة باستخدام القيم الفردية المناسبة.

مثال (٤-٧) :

يوضح هذا المثال كيفية تمثيل المتغيرات فى الحساب العلاقى (الخاص بمجموعات القيم المرتبة) لإيجاد أسماء الموردين Sname وأسماء قطع الغيار Pname الموردة وكمياتها. Qty كما هو مبين فى الشكل رقم (٤-٣٥) الذى يمثل الجدول العلاقى (r) الناتج.

شكل رقم (٤-٢٣) يمثل الجدول العلاقى (r) الناتج

Sname	Pname	Qty
r	r [2]	r [3]

١- الجداول المستخدمة هى: S . P . S P

٢- دع المتغير (4) x يمثل الجدول العلاقى S ، (5) y يمثل الجدول العلاقى P ، z (٣) يمثل الجدول العلاقى SP ، (3) r يمثل الجدول العلاقى الناتج.

٣- الصيغة التعبيرية :

$$\{r\,(3) \mid S\,(\chi) \wedge P(y) \wedge \mathrm{SP}\,(z) \wedge \chi[1] = z\,[1] \wedge y\,[1] = z\,[2] \wedge$$
$$r\,[1] = \chi\,[2] \wedge r[2] = y\,[2] \wedge r[3] = z\,[3]\}$$

٤- يعتبر هذا التعبير تعبيراً حراً. ولكن يتحول إلى تعبير مرتبط لو تم استخدام محددات القياس مثل :

$$\{r \mid (\exists\, x)\,(\exists\, y)\,(\exists\, z)\; .....\}$$

وتُبنى لغة الإنجريس Ingris Language على الحساب العلاقى لمجموعات القيم المرتبة والتى يمكن أن يعبر عنها بتعريف نطاق قيم المتغيرات.

(ب) الحساب العلاقى الخاص بنطاق القيم Domain Relutional Calculus :

يتم بناء الحساب العلاقى الخاص بنطاق القيم باستخدام نفس العوامل المستعملة فى الحساب العلاقى الخاص بالمجموعات القيم المرتبة. وتتمثل الاختلافات الرئيسية بينهما فى الآتى :

(١) لا توجد متغيرات لمجموعات القيم المرتبة فى الحساب العلاقى الخاص بنطاق القيم ، لكن توجد متغيرات لنطاق القيم لكى تمثل محتويات مجموعات القيم المرتبة. حيث يمثل كل متغير فى نطاق القيم لقيمة معينة فردية قيمة واحدة فقط، فى حين أن كل متغير لمجموعة قيم مرتبة يمثل مجموعة واحدة فقط فى الحساب العلاقى الخاص بمجموعات القيم المرتبة.

(٢) أى قيمة فردية هى قيمة من نطاق قيم الجدول العلاقى. على سبيل المثال : لو فرض أن الجدول العلاقى R ، فإن القيمة الفردية تكون *Xi* فى حالة ما إذا كان مخطط الجدول العلاقى R كالتالى:

$$R\,(X_1\ ,\ X_2\ ,\ X_3\ ,\ \ldots.\ ,X_n)$$

حيث إن :

$$1 \leq i \leq n$$

وهو جدول علاقى من الدرجة N-ary ، وكل xi هو ثابت أو متغير لنطاق القيم أو شرط معين مثل (x χ y) حيث إن x ، y هى ثوابت أو نطاق قيم او متغيرات و (_) عامل علاقى relational Operator. ومعنى القيمة الفردية (x χ y) هو أن x ، y يجب أن تمتلك القيم التى تجعل الشرط صحيحاً.

(٣) الصيغ داخل الحساب العلاقى الخاص بنطاق القيم تستخدم العوامل المنطقية كما هو الحال فى الصيغ داخل الحساب العلاقى الخاص بمجموعات القيم المرتبة.

(٤) أيضًا تستعمل محددات القياس $(\exists x)$ ، $(\forall x)$ لتشكل تعبيرات الحساب العلاقى الخاص بنطاق القيم ، ولكن (x) فى هذه الحالة تمثل متغير نطاق القيم بدلاً من متغير مجموعة القيم المرتبة .

مثال (٤-٨) :

يوضح هذا المثال كيفية تمثيل المتغيرات فى الحساب العلاقى (الخاص بنطاق القيم) لإيجاد أسماء الموردين Sname وأسماء قطع الغيار Pname الموردة وكمياتها. Qty ويمكن التعبير عن ذلك كالتالى :

$$\{\alpha bc \mid S(\chi_1, \alpha, \chi_3, \chi_4) \wedge P(y_1, b, y_3, y_4, y_5) \wedge SP(Z_1, Z_2, C) \wedge \chi_1 = Z_1 \wedge y_1 = Z_2\}$$

حيث إن :

a : تمثل اسم المورد Sname.

b : تمثل اسم قطعة الغيار Pname.

c : تمثل الكمية qty.

وهذا يعنى أن كلاً من c ، b ، a متغيرات خاصة بالنتائج. ولكى يتم تجنب أى التباس يجب تعريف كل متغير قبل استعماله ثم التعبير عنه من خلال استعماله.

صفات لغة الاستعلام المبينة على نطاق القيم :

لغة الاستعلام Query Language المبنية على دعم الحساب العلاقى الخاص بنطاق القيم ينبغى أن تصنف إلى ثلاثة أشكال هى:

(١) قائمة لغة الاستعلام menu :

أسلوب يفترض عدم معرفة المستخدم بكل الأوامر Commands فى القائمة ومن ثم ليس من الضرورى تذكر كل منها. كما أنها تتطلب استخدام قليل للوحة المفاتيح: مما يقلل الأخطاء، وإن كانت أقل تحكمًا فى النظام وعادة تعرض للمستخدم البسيط . وتكتب القائمة باللغة الانجليزية .

(٢) نماذج لغة الاستعلام Forms :

أسلوب يعبر عن [illegible] البيانات خلال نموذج محدد يتم ملؤه لتشكيل الاستعلام. وهو عادة يستخدم عمليات لخطوات معروفة ومحددة ويستخدم غالبًا فى العيادات الطبية ونظم المخازن وغيرها.

(٣) أوامر لغة الاستعلام Commands :

وهى تشبه لغة الاستعلام البنائية Structured Query Language (SQL). وتفترض معرفة كل الأوامر وفهم معانيها ومعرفة كيف ومتى يستخدم كل منها. وهى تزود المستخدم بالتحكم الكامل فى النظام ولكن بكتابة الأوامر وتذكرها.

ثالثاً - قواعد سلامة قاعدة البيانات العلاقية :

قواعد سلامة النموذج العلاقى تندرج تحت ثلاثة جزئيات هى:

- قواعد سلامة الكينونة entity أو النطاق Domain .
- قواعد السلامة داخل الجدول العلاقى Intra-relation .
- قواعد السلامة المرجعية referential integrity .

وتتم قواعد سلامة قاعدة البيانات من خلال المفاتيح الرئيسية Primary Keys والمفاتيح الخارجية Foriegn Keys ؛ لذا يجب تعريف كل من هذين المفهومين:

المفتاح الرئيسى Primary Key :

يُعَدُّ خاصية أو أقل مجموعة من الخصائص (الحقول) لا تتكرر قيمتها مع كل قيمة مرتبة (صف) لتعريف الجدول العلاقى relation.

المفتاح الخارجى Foreign Key :

يُعَدُّ خاصية (حقل) أو مجموعة من الخصائص (الحقول) فى الجدول العلاقى relation. إحداها ذات قيمة يتم استخدامها لعمل تعادل مع قيمة خاصية المفتاح الرئيسى فى جدول علاقى آخر ؛ لذلك فإن المفتاح الخارجى (مفتاح مشترك) والمفتاح

الرئيسى المناظر له الرئيسى فى الجدول العلاقى الآخر يجب أن يتم تعريفهما على نفس نطاق القيم.

١- قواعد سلامة الكينونة entity أو نطاق القيم Domain :

وتركز هذه القاعدة على المحافظة على قيم الخاصية (الحقل) داخل الجداول العلاقية. وتؤكد هذه القاعدة على أنه ينبغى ألا تكون قيم خصائص المفتاح الرئيسى أو أى قيم مرتبة (صف) tuple خالية من القيم Not Null. وأحيانًا تسمى هذه القاعدة قاعدة سلامة نطاق القيم Domain.

٢- قواعد السلامة داخل الجدول العلاقى :

ترتبط قواعد السلامة داخل الجدول العلاقى بصحة العلاقة بين خصائص (الحقول) نفس الجدول العلاقى. ومنها على سبيل المثال: التبعيات الوظيفية Functions dependancies ، وكذلك المحافظة على عدم تكرارية المفتاح.

٣- قواعد السلامة المرجعية Referential Integrity Rules :

تركز قواعد السلامة المرجعية على المحافظة على صحة وعدم تضارب (توافق) علاقات الربط بين الجداول العلاقية. وتتضمن القواعد التالية:

(١) قاعدة الإضافة :

حفاظًا على قواعد السلامة المرجعية لا يجوز إضافة مجموعة قيم مرتبة فى جدول علاقى (نوع كينونة ابن) ما لم توجد مجموعة قيم مرتبة ملائمة فى الجدول العلاقى المرتبط به (نوع كينونة أب).

(٢) قاعدة الحذف :

توجد ثلاثة خيارات لقاعدة الحذف يتم الخيار من بينها عند إنشاء الجدول العلاقى (نوع الكينونة) :

أ - لا يجوز حذف مجموعة قيم مرتبة (كينونة) من جدول علاقى (نوع كينونة أب) إذا كانت هناك مجموعة قيم مرتبة (كينونة) أو أكثر ترتبط بها فى جدول علاقى آخر (نوع كينونة ابن).

ب - جعل قيمة المفتاح الخارجى فى مجموعة القيم المرتبة (الكينونة الابن) غير معروفة ثم تنفيذ عملية حذف على مجموعة القيم المرتبة (الكينونة الأب).

ج - حذف على مجموعة القيم المرتبة (الكينونة الأب) وجميع مجموعات القيم المرتبة المرتبطة بها فى الجداول العلاقية الأخرى.

التبعيات الوظيفية Functional Dependencies (FDs) :

التبعية الوظيفية هى ارتباط خاص بين خاصتين أو أكثر فى الجدول العلاقى. إحدى هذه الخصائص تابعة وظيفيًا للخاصية الأخرى[٤]. على سبيل المثال : فى حالة افتراض وجود جدول علاقى وليكن R ، ويحوى مخططه خاصيتين ، هما: الخاصية (B)التابعة وظيفيًا للخاصية (A). بمعنى أن كل قيمة فى الخاصية (A) تحدد قيمة واحدة فى الخاصية (B). ويمكن التعبير عن تلك التبعية الوظيفية كالآتى:

$$A \rightarrow B$$

الطريقة الوحيدة لتحديد التبعيات الوظيفية هى تحديد مخطط الجدول العلاقى ومعرفة ما تعنيه الخصائص. ففى المثال (٤-١٠) تم إيضاح أن خاصية اسم المورد S تستطيع وظيفياً أن تحدد خاصية عنوان المورد A : لأنه لا يمكن تخزين عنوانين لمورد واحد فى قاعدة البيانات. ففى مخطط الجدول العلاقى SS (S . A . I, P) توجد التبعيات الوظيفية التالية:

$$S \rightarrow A$$

$$SI \rightarrow P$$

تسمى خصائص الطرف الأيسر للتبعية الوظيفية بالمحدد Determinant

بديهيات التبعيات الوظيفية Axioms for FDS :

* إذا كانت فئة التبعية الوظيفية F تتضمن منطقياً :

$$X \rightarrow Y$$

فإنها يمكن التعبير عنها كالآتى :

$$F : X \rightarrow Y$$

* فئة التبعيات الوظيفية التى تتضمن بشكل منطقى التبعية الوظيفية F تسمى بالفئة الانغلاقية F clusure ويرمز لها بالرمز F^+. ويمكن التعبير عنها كالتالى :

$$F =_+ \{X \rightarrow Y \mid F : X \rightarrow Y\}$$

- لتحديد المفاتيح الأساسية والشاملة يلزم حساب فئة التبعيات الوظيفية الإنغلاقية F^+ من فئة التبعية الوظيفية F وهذا يتطلب اتباع قواعد استنتاج التبعيات الوظيفية . حيث إن هذه القواعد تسمح باستنتاج كل التبعيات الوظيفية فى فئة التبعيات الوظيفية الإنغلاقية F^+.

- تسمى مجموعة القواعد الخاصة باستنتاج التبعيات الوظيفية باسم بديهيات أرمسترونج Armstrong's axioms.

- يمثل مخطط الجدول العلاقى مع فئة الخصائص الفئة الشاملة Universal set والتى يرمز لها بالرمز U. على سبيل المثال الفئة الشاملة للجدول العلاقى R وفئة التبعيات الوظيفة التى تتضمن فقط الخصائص فى الفئة الشاملة U يمكن أن يرمز لها بالرمز UR.

- سوف يشار إلى كل خاصية بالحروف الأولى منها (... A. B. C) ويشار لأكثر من خاصية (فئة جزئية) بالحروف الأخيرة (مثل Z. Y. X).

- سوف يشار إلى أسماء العلاقات بالحروف الكبيرة فى حين سوف يشار إلى القيم (الوقائع) بالحروف الصغيرة.

قواعد استنتاج التبعيات الوظيفية Inference Rules :

لو فرض أن X . Y . W . Z هى مجموعة الخصائص التى يتشكل منها مخطط جدول علاقى معين ، فان قواعد استنتاج التبعيات الوظيفية تتضمن القواعد التالية:

١- قاعدة الارتداد Reflexivity :

لو كانت X فئة جزئية من الفئة الشاملة U ، والتى يمكن أن يرمز لها جبريًا كالأتى :

$$X \subseteq U$$

وأن Y فئة جزئية من الفئة X ، أى أن :

$$Y \subseteq X$$

فإن فئة التبعية الوظيفية F هى :

$$F : X \rightarrow Y$$

على سبيل المثال :

لو أن A هى فئة جزئية مناسبة من AB ، أى أن :

$$A \subset AB$$

فإن الفئة الجزئية المناسبة A تعتمد على AB. وبمعنى آخر الفئة الجزئية AB تحدد الفئة الجزئية المناسبة A. ، والتى يمكن أن يعبر عنها جبريًا كالآتى:

$$AB \rightarrow A$$

٢- قاعدة الازدياد Augmentation :

لو فرض أن :

$$X \rightarrow Y$$

$$Z \subseteq U$$

فإن فئة التبعية الوظيفية F هى :

$$F : XZ \rightarrow YZ$$

حيث إن X . Y . Z هى مجموعة من الخصائص ، وأن التبعية الوظيفية X _ Y قد تكون فى فئة التبعية الوظيفية F.

٣- قاعدة الانتقالية Transitivity :

لو أن التبعيات الوظيفية هى:

$$X \rightarrow Y$$
$$Y \rightarrow Z$$

فإن التبعية الوظيفية طبقًا لقاعدة الانتقالية هى:

$$X \rightarrow Z$$

٤- قاعدة الاتحاد Union :

لو أن التبعيات الوظيفية هى:

$$X \rightarrow Y$$
$$Y \rightarrow Z$$

فإن التبعية الوظيفية طبقًا لقاعدة الاتحاد هى:

$$X \rightarrow YZ$$

٥- قاعدة الانتقالية الزائفة Pseudotransitivity :

لو أن التبعيات الوظيفية هى:

$$X \rightarrow Y$$
$$WY \rightarrow Z$$

فإن التبعية الوظيفية طبقًا لقاعدة الانتقالية الزائفة هى:

$$XW \rightarrow Y$$

٦- قاعدة التفكيك Decomposition :

لو أن التبعيات الوظيفية هى:

$$X \rightarrow Y$$
$$Z \subseteq Y$$

فإنه طبقًا لقاعدتى الارتداد و قاعدة الانتقالية يمكن الحصول على التبعية الوظيفية التالية:

$$X \rightarrow Z$$

المفاتيح Keys :

المفتاح هو خاصية أو مجموعة خصائص تعرف الجدول العلاقى بشكل لا يتكرر .

لو فرض أن: مخطط الجدول العلاقى R (A1 , A2, ... An) ، فإن التبعيات الوظيفية F ، و X فئة جزئية من مخطط الجدول العلاقى. أى أن:

$$X \subseteq \{A1 ,A2, ..., An)$$

فإن X تصبح مفتاحاً لو أن التبعية الوظيفية X تحدد كل الخصائص هى فى فئة التبعيات الوظيفية الانغلاقية +F . وهذا يعنى أن X هى المفتاح الشامل Super Key لأن كل الخصائص تعتمد عليه.

المفتاح المرشح Candidate Key :

هو أقل مجموعة خصائص التى تحدد وظيفياً كل الخصائص ولكن لم يتم اختيارها لكى تكون مفتاحاً اساسياً.

المفتاح الشامل Super Key :

هو المفتاح الذى يشمل معظم الخصائص وجزء منه يمكن أن يصف الجدول العلاقى: وذلك لأن المفتاح الأساسى جزء من المفتاح الشامل.

أقل فئة للتبعيات Minimal Set of dependencies :

١- ينبغى أن يكون الجانب الأيمن لفئة التبعية الوظيفية F هو خاصية واحدة أو التأكيد من أنه لا توجد خصائص فى الجانب الأيمن زائدة.

٢- فى حالة عدم وجود التبعية الوظيفية.

$$X \rightarrow A$$

فى فئة التبعية الوظيفية F تكون الفئة :

$$F - \{X \rightarrow A\}$$

مكافئة لفئة التبعية الوظيفية F الجديدة. وبذلك يتم ضمان أنه لا توجد تبعية وظيفية فى فئة التبعية الوظيفية F تكون زيادة.

٣- وهكذا بدون الخاصية A فى F والفئة الجزئية المناسبة Z للخاصية X تكون:

$$F - \{X \rightarrow A\} \cup \{Z \rightarrow A\}$$

تكافىء فئة التبعية الوظيفية F الجديدة وبذلك يتم ضمان أنه لا خصائص فى الجانب الأيسر تكون زيادة.

حساب الفئة الانغلاقية Computing Closure :

فى حساب فئة التبعيات الوظيفية الانغلاقية F^+ لمجموعة تبعيات الوظيفية F تكون مهمة الوقت المستنفد عمومًا بسيطة؛ وذلك لأن مجموعة التبعيات الوظيفية فى فئة التبعيات الوظيفية الانغلاقية F^+ قد تكون ضخمة حتى لو أن لمجموعة تبعيات الوظيفية F فى حد ذاتها كانت صغيرة[٤].

لو أخذ فى الحسبان أن فئة التبعيات الوظيفية هى :

$$F = \{A \rightarrow B_1, A \rightarrow B_2, ... A \rightarrow B_n\}$$

فإن فئة التبعيات الوظيفية الانغلاقية +F تتضمن كل التبعيات الوظيفية :

$$A \rightarrow y$$

حيث إن Y هى الفئة الجزئية $\{B1, B2, ... ,Bn\}$. وأن حساب الخاصية الانغلاقية X^+ لمجموعة الخصائص X تأخذ وقتًا نسبيًا حسب طول التبعيات الوظيفية فى فئة التبعيات الوظيفية F.

مثال (٤-١١) :

لتوضيح كيفية حساب أقل فئة للتبعيات الوظيفية نفرض أن مخطط الجدول العلاقى R يحتوى على مجموعة من الخصائص هى {A, B, C, D, G} إلى جانب مجموعة من التبعيات الوظيفية هى :

$$R (ABCDEG = \{AB \rightarrow C, C \rightarrow A, BC \rightarrow D, ACD \rightarrow D, D \rightarrow EG,$$

$$BE \rightarrow C, CG \rightarrow BD, CE \rightarrow AG\}$$

الخطوات :

$X_0 = BD$

$X_1 = BDEG \quad (D \rightarrow EG)$

$X_2 = BCDEG \quad (BE \rightarrow C, D \rightarrow EG)$

$X_3 = ABCDEG \quad (C \rightarrow A, BC \rightarrow D, D \; EG,$

$CG \rightarrow BD, CE \rightarrow AG)$

ومن هنا يتبين أن أقل فئة تبعية انغلاقية هى:

$(BD)^+ = ABCDEG.$

سمات نموذج البيانات العلاقى :

١- لغة الاستعلام تتعامل مع مجموعات القيم المرتبة فى نفس الوقت؛ لذا تسمى لغة استعلام عالية المستوى. وهى تمكن المستخدم من إضافة وحذف وتعديل مجموعات القيم المرتبة (السجلات) Tuples [٨] .

٢- العديد من اللغات يعبر عن العمليات الخاصة بنطاق القيم Domain ومجموعات القيم المرتبة (الصفوف) tuples . منها الحساب العلاقى ولكنه ليس لغة إجرائية. فى حين أن الجبر العلاقى هو لغة إجرائية تكافىء قوتها الحساب العلاقى[١].

٣- تعتمد على الأساس العلمى لنظرية الفئات الرياضية التى تميزها عن النماذج التبحرية (الهرمية - الشبكية).

محدودية النماذج العلاقية :

١- أنه يستغل تجانس البيانات أفقياً ورأسياً : لأنه يمثلها كصفوف.

٢- يجب أن تكون البيانات فى شكلها الطبيعى الأول 1NF الذى يعترض التمثيل المباشر للحقول (الخصائص) متعددة القيم Valued multi .

٣- كينونة العالم الحقيقى لا تتناسب مع النموذج العلاقى مباشرة؛ لذلك فإن التفكيك الصناعى أصبح ضرورياً، وهذا التفكيك يقلل الكفاءة ويرفع إمكانية فقدان الرابطة المنطقية للبيانات[٩].

الهوامش :

1- [KORTH, 1986], Henry F. Korth and Abraham Silberschatz, **Database System Concepts**, International Edition, McGraw-Hill, Inc., 1986.

2- [ULLMAN,1988], Ullman J.D., **Principles of Database and Knowledge-Base System**, Vol. 1, Computer Science Press, 1988.

3- [DATE,1990], C.J. Date, An Introduction to Database Systems, Volume I, Fifth Edition, Addison-Wesley Publishing Company Inc.,1990.

4- [ELMASRI, 1989], Ramez Elmasri and Shamkant B. Navathe, **Fundamentals of Database Systems**, Benjamin/Cummings, Redwood City, Calif., 1989.

5- [Connolly 1996], Thomas. M. Connolly and Carolyn E. Begg, **Database Systems**, Addison-Wesley Publishing Co., Inc., 1996.

6- [Mcallister 1998], Mcallister, Andrew J. and Shorpe David, An Approach for Decomposing N-ary Data Relationships , **Software Practice and Experience**, Vol. 28(2), Feb.,1998.

7- [DATE,1995], C.J. Date, **An Introduction to Database Systems** Volume I, Sixth Edition, Addison-Wesley Publishing Company Inc.,1995.

8- [GRANT, 1987], John Grant, **Logical Introduction to Databases**, Harcourt Brace Jovanovich, Publishers and its subsidiary, Academic Press, 1987.

9- [Brathwaite 1991], Dr. Kenmore S. Brathwaite, **Relational Databases, Concepts, Design, and Administration**, McGraw-Hill Inc.New York,1991.

الفصل الخامس

قاعدة البيانات العلاقية جيدة التصميم

مقدمة :

من الأهمية بمكان فى هذا الفصل البدء بدراسة وتحليل تبعية البيانات ، ويرجع السبب وراء ذلك إلى أن التصميم الجيد لقاعدة البيانات يؤدى إلى حماية تبعية البيانات فى اتصال قاعدة البيانات الحقيقية. وسوف يتم مناقشة الموضوعات التالية:

تبعية البيانات :

من الأهمية البالغة لمصممى قواعد البيانات العلاقية دراسة وتحليل تبعية البيانات، وذلك لتمييز بين قاعدة البيانات ذات التصميم الجيد وقاعدة البيانات ذات التصميم الرديء التصميم. ويسهل عمل هذه الدراسة عن طريق معرفة الروابط بين خصائص الجداول العلاقية. كما سيتم تقديم تلخيص لمعالم كل من:

- قواعد البيانات الرديئة التصميم .
- قواعد البيانات الجيدة التصميم .

التفكيك :

. التفكيك هو العملية التى بها يتم تحليل شكل قاعدة البيانات وتجزيئها إلى جداول علاقية أبسط. وقد يؤدى التفكيك إلى فقدان جزء من المعلومات.

الربط :

هناك نوعان من الربط. أحدهما الربط بدون فقدان، وهذا يعنى أن كل جدول علاقى فى المخططات العلاقية يحقق التبعيات الوظيفية عند عمل ربط لها. بخلاف ذلك يكون الربط فقدانيًا. كما سيتم التعرف على أهداف التفكيك والتى تتمثل فى حماية كل من المعلومات والقيود.

الطرق العلمية لتصميم قاعدة البيانات العلاقية :

كان من الضرورى استعراض الطرق العلمية الخاصة بتصميم قاعدة البيانات العلاقية. ومن هذه الطرق ما يلى:

- الطريقة التحليلية.
- الطريقة الاصطناعية.
- الطريقة الدلالية.

الطريقة التحليلة للتصميم :

تتضمن هذه الطريقة مجموعة من الخطوات التى تشتمل على التعامل مع الفئة الشاملة U ، ومعرفة الشكل الطبيعى الذى تنتمى إليه الجداول العلاقية، والتأكد من صفة التفكيك.

التطبيع :

يقصد بالتطبيع الوصول بالجدول العلاقى إلى شكل طبيعى معين. ولكل شكل طبيعى مجموعة من الشروط التى تحدد حالته، سواء كان فى الشكل الطبيعى الأول ، والثانى ، والثالث ، والرابع ، والخامس أم فى الشكل الطبيعى الخاص بيويس كود.

الطريقة الاصطناعية للتصميم :

تتضمن هذه الطريقة مجموعة من الخطوات التى تشتمل على التعامل مع الفئة الشاملة U وقراءة الحروف، والاعتماد على الرسومات الخاصة بالتبعيات.

مقدمة فى لغة الاستعلام البنائية :

سوف يتم عرض مدخل مبسط بأمثلة توضيحية للغة الاستعلام البنائية ومميزات استخدامها: لما تحظى به من انتشار واسع النطاق. وتتكون هذه اللغة من تعليمات خاصة لتعريف البيانات، ومعالجتها والتحكم فيها.

تبعية البيانات Data Dependency :

تصف تبعية البيانات اعتماد البيانات بعضها على بعض: لذا ينبغى أن تكون الروابط بين الخصائص فى الجداول العلاقية ذات معنى، ويتم ذلك عن طريق التحكم فى علاقات الربط باستخدام المفاتيح الأساسية التى لا تتكرر فى كل كينونة، إلى جانب ذلك ينبغى أن توضع قيود داخل الجداول العلاقية.

مثال (٥-١) :

يبين هذا المثال تبعية البيانات كما هو موضح بالجدول العلاقى SS الخاص ببيانات المورد فى الشكل رقم (٥-١). حيث إن كل مورد Supplier (S) يورد مواد معينة Items (I) ، وكل مورد له عنوان Address (A) وكل مادة لها سعر Price (P).

شكل رقم (٥-١) الجدول العلاقى *SS* الخاص ببيانات المورد

SS

S	A	I	P
Mahmoud	Dammam	Paper	25
Mahmoud	Dammam	Books	15
Ibrahim	A1- Khobar	Paper	10
Mahmoud	Dammam	Scripts	7

وتبدو فى هذا المثال مشكلة البيانات الزائدة عن الحد redundancies أكثر وضوحاً حيث إن زيادة الطلب على توريد مواد مختلفة (I) لنفس المورد يتطلب تخزيناً متعدداً لنفس اسم المورد (S) وعنوانه (A) أيضاً. وكذلك عندما تتغير خاصية المادة (I) الزائدة لا بد من تغيير كل الجداول العلاقية relations بقاعدة البيانات.

ومن الأهمية بمكان أن يتم التنويه إلى نقطة مهمة هى مشكلة الحذف الشاذ deletion anomaly التى تظهر عندما تحذف مادة معينة من العلاقة SS . فى هذه الحالة سوف يتم حذف كل مجموعة القيم المرتبة tuple الخاصة بها بدلاً من حذف المادة ، ومن ثم يتم حذف ذلك المورد (S).

قاعدة البيانات الرديئة التصميم Bad-Designed DB :

يؤدى عدم الالتزام بقيود السلامة المرجعية referential integrity إلى انتهاك سلامة قاعدة البيانات. وقيد السلامة المرجعية هو قاعدة عمل تحافظ على سلامة مرجعية كينونة لكينونة أخرى أو أكثر فى قاعدة البيانات. وهناك أربعة مشاكل فعلية تواجه تصميم قاعدة البيانات وتتمثل معظمها فى الأخطاء. والأخطاء (anomalies) هى تضاربات قد تنتج عند تحديث جدول علاقى به بيانات متكررة (زائدة عن الحد). وهذه المشاكل التى تؤدى إلى ردائة قاعدة البيانات هى :

١- البيانات الزائدة عن الحد redundancy .

٢- التعديل الخطأ (الشاذ) Update anomaly لتغيير مكونات مجموعة قيم مرتبة (صف واحد فقط).

٣- الإضافة الخطأ (الشاذة) Insertion anomaly لإضافة مجموعة قيم مرتبة (صف كامل).

٤- الحذف الخطأ (الشاذ) Deletion anomaly لحذف مجموعة قيم مرتبة (صف كامل).

وتظهر هذه المشاكل فى حالة عدم مراجعة البيانات الزائدة عن الحد. وللوصول إلى التصميم الصحيح لقاعدة البيانات يجب معرفة :

- أسماء الخصائص.
- الروابط بين الخصائص (التبعية).
- المفتاح الأساسى للجدول العلاقى.

مثال (٥-٢) :

هناك العديد من المشاكل فى المثال (٥-١) التى ينبغى إيجاد حل ملائم لها. وفيه تم الاحتفاظ بالجدول العلاقى ذى المخطط التالى:

SS (S , A , I , P)

ومن هذه المشاكل:

١- زيادة البيانات عن الحد.

يتم تكرار عنوان المورد (A) واسم المورد (S) فى حالة توريد كل مادة (I) .

٢- التعديل الشاذ :

تتبعًا لزيادة البيانات عن الحد ، فإن تعديل عنوان مورد (A) ما فى إحدى مجموعات القيم المرتبة يسهل تحديثه. فى حين لم يتم التحديث فى بقية الصفوف المرتبطة بذلك المورد. ومن ثم لن يتم الحصول على عنوان واحد للمورد بل أكثر من عنوان.

٣- الإضافة الشاذة :

تظهر هذه المشكلة فى حالة إضافة عنوان لمورد (A) ما لم يكن هذا المورد يورد مادة واحدة على الأقل ؛ ولذا فمن الصعب الاحتفاظ بأسماء الموردين وعناوينهم فقط.

٤- الحذف الشاذ :

تبدو هذه المشكلة واضحة عندما يتم حذف كل المواد الموردة بواسطة مورد معين ؛ لأنه فى هذه الحالة سوف تفقد كل المعلومات الخاصة بذلك المورد وفيها عنوانه. وعلاج هذه المشاكل بهذا المثال تكمن فى وضع الموردين فى جدولين علاقيين يمكن توضيحهما بمخططين علاقيين كالآتى:

SA (S , A)

SIP (S , I , P)

وبإلحاق الوقائع بمخططات الجداول العلاقية السابقة يمكن الحصول على الجداول العلاقية فى الشكل (٥ – ٢).

شكل رقم (٥-٢) للجدولين العلاقيين SA , SIP

SA

S	A
Ibrahim	A1- Khobar
Mahmoud	Dammam

SIP

S	I	P
Mahmoud	Paper	25
Mahmoud	Books	15
Ibrahim	Paper	10
Mahmoud	Scripts	7

حيث إنه فى الجدول العلاقى SA تم وضع المورد (S) ، عنوانه (A) ؛ ومن ثم لا توجد بيانات زائدة عن الحد. وبناء عليه يمكن إضافة عنوان لأى مورد جديد حتى لو لم يورد أى مواد. أما فى الجدول العلاقى SIP فإن السعر يتغير مع تغير كل مادة. ومع ذلك، يجب استعمال الربط بين الجدولين العلاقيين السابقتين.

قاعدة البيانات الجيدة التصميم Well-designed DB :

تكون قاعدة البيانات جيدة التصميم لو تم تحقيق النقاط التالية:

١- شكل طبيعى معين Certain Normal Form :

يجب أن تكون قاعدة البيانات فى شكل طبيعى معين ، كلما كانت فى الشكل الطبيعى الأعلى، سوف تكون أفضل ؛ لأن كل شكل طبيعى يتجنب مجموعة مشاكل معينة.

٢- حماية التبعيات preservation of dependencies :

يجب أن يحمى التصميم تبعيات البيانات فى اتصال قاعدة البيانات الحقيقية.

٣- تفكيك بدون فقدان.

التفكيك :

إنها العملية التى بها يتم تحليل شكل قاعدة البيانات وتفكيكه إلى جداول علاقية أبسط لكى تصل إلى الشكل المرغوب فيه (الشكل الطبيعى الأعلى). فى حالة عدم فقدان أى معلومات أثناء عملية التفكيك تسمى "تفكيك بدون فقدان" بخلاف ذلك تسمى "تفكيك فقدانى" Loosely decomposition. وعملية التفكيك بدون فقدان تكافىء الاختيار الرأسى فى الجبر العلاقى؛ حيث إن اتحاد الاختيار الرأسى سوف يؤدى إلى الجدول العلاقى الأصلى.

التفكيك الفقدانى Loosely decomposition :

يؤدى إلى فقدان كمية معلومات ودائمًا يكون فقدان جزء من معلومات الجدول العلاقى الأصلى.

التفكيك بدون فقدان Lossless decomposition :

أى جدول علاقى يتم تفكيكه إلى جداول علاقية متعددة يجب أن يتم حماية المعلومات الأصلية الموجودة بداخله ، بخلاف ذلك سيُفقد جزء من معلوماته الأصلية.

أسباب تفكيك مخطط الجدول العلاقى :

أحد الدوافع وراء تنفيذ تفكيك مخطط الجدول العلاقى هو أن التفكيك تجنب بعض المشاكل التى يؤدى إلى جعل قاعدة البيانات رديئة. على سبيل المثال لو تم أخذ مخطط الجدول العلاقى SS فى الحسبان ، وكانت التبعيات الوظيفية الخاصة به هى:

$$S \rightarrow A$$

$$SI \rightarrow P$$

لو أن قاعدة البيانات استعملت الجدولين العلاقيين SA ، SIP بدلاً من الجدول العلاقى SS ، فإنه من الطبيعى أن يكون من المتوقع أن الجداول العلاقية الحالية لمخططى هذين الجدولين العلاقيين هى اختيار رأسى لمخطط الجدول العلاقى SS.

الربط :

الربط بدون فقدان Lossless Join :

لو أن R هى مخطط الجدول العلاقى المفكك إلى المخططات $A_1, A_2, ..., A_n$. وأن D هى مجموعة التبعيات الوظيفية. فإن التفكيك فيما يتعلق بمجموعة التبعيات الوظيفية D يكون تفكيك ربط بدون فقدان ، حيث إن كل جدول علاقى فى المخططات علاقية ($A_1, A_2, ..., A_n$) يحقق التبعيات الوظيفية عند عمل ربط لها. ولا بد من عمل اختبار للربط بدون فقدان للتأكد من أن الربط كان بدون فقدان للتبعيات الوظيفية أم لا. وفى أى قاعدة بيانات ، ينبغى أن تكون صفوف القيم فى الجانبين الأيمن والأيسر متساوية.

خطوات التأكد من الربط بدون فقدان :

المدخلات :

- مخطط الجدول العلاقى $R = A_1, A_2, ... A_n$
- مجموعة التبعيات الوظيفية F والتفكيك حيث إن :

$$R = R_1 \cup R_2 U .... UR_K$$

المخرجات : قرار بخصوص ما إذا كان التفكيك بدون فقدان أم لا .

الطريقة :

١- يتم إنشاء الجدول العلاقى الذى يحتوى على عدد (n) من الخصائص (أعمدة) ، وعدد (K) من مجموعات القيم المرتبة (صفوف).

٢- يتم وضع الرمز aj فى الصف (i) والعمود (j) لو كانت الخاصية Aj موجودة فى الجدول العلاقى Ri. بخلاف ذلك يتم وضع الرمز bij .

٣- يتم تكرار ذلك على اعتبار أن كل تبعية وظيفية فى فئة التبعيات الوظيفية F هى:

$$X \rightarrow y$$

حتى لا يمكن عمل أى تغيير على الجدول العلاقى.

٤- فى كل مرة لابد من الأخذ فى الحسبان توافق التبعية فى كل الأعمدة مع ملاحظة الصفوف التى توافق تلك التبعية فى هذه الأعمدة. واستبدال الرمز bij بما يتوافق مع التبعية بالرمز aj.

٥- لو حصلنا على صف واحد على الأقل يحتوى على a' S فإن الربط يكون بدون فقدان للتبعيات الوظيفية ، وبخلاف ذلك يكون الربط فقدانياً.

مثال (٥-٣) :

فى حالة تفكيك الجدول العلاقى SS الموجود بالمثال (٥-٢) إلى الجدولين العلاقيين SA , SIP. حيث إن التبعيات الوظيفية هى:

$$S \rightarrow A$$

$$SI \rightarrow P$$

فإن الجدول العلاقى المبدئى يكون SA حيث إن رمزى الصفين (a1' S) يتوقفان على الخاصية S على سبيل المثال. وإن الصفين R2 و R1 يمثلاين التبعيتين الوظيفيتين السابقتين على التوالى. وإن الخاصية التى ليس لها تمثيل فى التبعية الوظيفية الخاصة بها يعبر عنها بالحروف b ويمثل الرمز الرقمى الصف ثم العمود. فى حين أن الخاصية التى لها تمثيل فى التبعية الوظيفية فيعبر عنها بالحرف a مع الأخذ فى الحسبان رقم العمود فقط كما هو موضح بالشكل رقم (٥-٣).

الشكل رقم (٥-٣) الجدول العلاقى المبدئى لتمثيل التبعيات الوظيفية

SA

	S	A	I	P
R1	a_1	a_2	b_{13}	b_{14}
R2	a_1	b_{22}	a_3	a_4

ويعمل التوازن بين حروف a' S فإن b_{22} تصبح a_2 ؛ لذا فإن الجدول الناتج كما هو موضح بالشكل رقم (٥-٤).

الشكل رقم (٥-٤) الجدول العلاقى النهائى لتمثيل التبعيات الوظيفية

SIP

	S	A	I	P
R1	a_1	a_2	b_{13}	b_{14}
R2	a_1	a_2	a_3	a_4

وحيث إنه تم الحصول على صف واحد (على الأقل) يحتوى على a's فإن الربط يكون بدون فقدان.

مثال (٥-٤) :

فى الجدول العلاقى SS لو افترضنا أن التبعيات الوظيفية كانت كالآتى:

$$S \rightarrow I$$

$$SA \rightarrow P$$

وأن المخططات الخاصة بالتبعيات الوظيفية هى SI والتى تمثل بالصف SAP ، R1 والتى تمثل بالصف R2 ؛ ولـذا فإن الجدول المبدئى يتم توضيحه بالشكل رقم (٥-٥).

الشكل رقم (٥-٥) الجدول العلاقى المبدئى لتمثيل التبعيات الوظيفية

SI

	S	A	I	P
R1	a_1	a_{12}	b_3	b_{14}
R2	a_1	a_2	a_{23}	a_4

وبفحص التبعيات الوظيفية نجد أنه يمكن استبدال b12 بالأمر a2

الشكل رقم (٥-٦) الجدول العلاقى النهائى لتمثيل التبعيات الوظيفية

SAP

	S	A	I	P
R1	a_1	a_2	a_3	b_{14}
R2	a_1	a_2	b_{23}	a_4

وحيث إنه لم يتم الحصول على أى صف يحتوى على رموز (a'S) فإن الربط فقدانى.

أهداف التفكيك :

بغض النظر عما إذا كان الجدول المفكك يحقق الشكل الطبيعى الثالث أو الرابع أو الخامس ، فهناك هدفان للتفكيك[٦].

* حماية المعلومات.
* حماية القيد.

(١) حماية المعلومات :

تعنى أن الجداول العلاقية المفككة ينبغى أن تكون قادرة وهى مجمعة على تخزين نفس مجموعة الحقائق كما فى الجدول العلاقى الأصلى. ويشار إلى هذه القدرة كصفة الربط بدون فقدان للتفكيك.

(٢) حماية القيـد :

تعنى أن القيود فى الجداول العلاقية الأصلية والمفككة ينبغى أن تتساوى. ويشار إلى هذا الهدف كصفة لحماية التبعية.

الطرق العلمية لتصميم قاعدة البيانات العلاقية :

هناك ثلاثة طرق لتصميم قاعدة البيانات ، هى:

١- الطريقة التحليلية Analytical approach .

٢- الطريقة الاصطناعية Aynthetical approach .

٣- الطريقة الدلالية Semantical approach .

تعمل الطريقتان الأولى والثانية بشكل جيد ، فى حين تعتبر الطريقة الثالثة حديثة ومازالت فى طور التطوير وإن كانت أصولها قديمة جداً.

الطريقة التحليلية للتصميم Analytical approach :

وتتضمن هذه الطريقة فى تصميم قاعدة البيانات الخطوات التالية :

(١) التعامل مع الفئة الشاملة U كجدول علاقى واحد للحصول على قاعدة البيانات.

(٢) التأكد من أن الجداول العلاقية فى الشكل الطبيعى بويس – كود (BCNF) لقاعدة البيانات الجديدة.

(٣) التأكد من أن التفكيك يتضمن الصفتين التاليتين :-

* عدم فقدان أى من التبعيات الوظيفية فى الفئة F .

* عدم فقدان أى من الخصائص أو البيانات.

التطبيع Normalization :

أنواع الخصائص :

فى المثال رقم (٤-٧) فى الفصل الرابع نجد أن مخطط الجدول العلاقى S الخاص بالمورد supplier هو [٧] : S (S# , Sname , Saddr , Status , City) يلحظ أن خاصية رقم المورد S# تمثل المفتاح، بينما فى مخطط الجدول العلاقى SP يتكون المفتاح من خاصيتين كالآتى:

SP (S# , P# , Qty)

وأيًا كانت الخصائص المكونة للمفتاح Key ، فإن كل خاصية تنتمى للمفتاح تسمى خاصية أولية .Prime أما الخاصية التى لا تنتمى إلى المفتاح تسمى باللاأولية Non-Prime .

أنواع التبعيـــة :

يبين الشكل رقم (٥-٧) ثلاثة جداول علاقية ضمن الفئة الشاملة U. وإن مجموعة التبعيات الوظيفية F هى [٥]:

$$X \rightarrow Y$$

$$S \rightarrow Y$$

Z

b

X

a	b

Y

c	b

الشكل رقم (٥-٧) الفئة الشاملة U للجداول العلاقية Z , X , Y

يبين الشكل رقم (٥-٧) أن الجدول العلاقي Z هو فئة جزئية مناسبة من الجدول العلاقى X. وأن الجدول العلاقى Y يعتمد جزئيًا على كل من الجدول العلاقى X و Z. بمعنى أن Y تعتمد فقط على خاصية واحدة فقط فى الفئة الانغلاقية F^+ . وهذا يمثل التبعية الجزئية Portial dependecy. أما لو كان الجدول العلاقى Z ليس فئة جزئية مناسبة من العلاقة X فإن الجدول العلاقى y سيصير تبعية كاملة Fully dependeney .

الشكل الطبيعى الأول First Normal Form (1NF) :

يكون الجدول العلاقى فى الشكل الطبيعى الأول (NF) إذا - ليس غير- كانت كل قيمة فى نطاق القيم لكل خاصية قيمة واحدة فقط غير قابلة للتجزئة ولذا فإن الجدول العلاقى يكون فى الشكل الطبيعى الأول إذا كان لا يحتوى على مجموعات متكررة. وتكون قاعدة البيانات فى الشكل الطبيعى الأول لو كان كل جدول علاقى بها فى الشكل الطبيعى الأول. ويبين الشكل رقم (٥-٨) الجدول R1 ليس فى الشكل الطبيعى الأول.

شكل رقم (٥-٨) الجدول العلاقى R1 ليس فى الشكل الطبيعى الأول (1NF)

RI

S#	Snane	Status	Part	Pcity
R1	Ahmed	20	X1 X2	Paris London
R2	Mohamed	30	X3 X5	Ryiadh Geddeh
R3	Ibrahim	35	X3 X4	Gairo Paris

ولتحويل الجدول العلاقى فى الشكل رقم (٥-٨) إلى الشكل الطبيعى الأول (NF١) لابد من إزالة المجموعات المتكررة كما هو موضح بالشكل رقم (٥-٩).

شكل رقم (٥-٩) الجدول العلاقى R2 فى الشكل الطبيعى الأول (1NF)

R2

S#	Snane	Status	Part	Pcity
S1	Ahmed	20	X1	Paris
S1	Ahmed	20	X2	London
S2	Mohamed	30	X3	Ryiadh
S2	Mohamed	30	X5	Geddeh
S3	Ibrahim	35	X3	Gairo
S3	Ibrahim	35	X4	Paris

الشكل الطبيعى الثانى Secand Normal Form (2NF) :

يكون الجدول العلاقى فى الشكل الطبيعى الثانى (2NF) لو كان فى الشكل الطبيعى الأول وكل خاصية لاأولية non_prime فى الجدول العلاقى تكون لها تبعية كاملة على المفتاح. بمعنى آخر لو كان للخاصية اللاأولية تبعية جزئية ، فإن الجدول العلاقى لا يكون فى الشكل الطبيعى الثانى. وتكون قاعدة البيانات فى الشكل الطبيعى الثانى لو كانت كل الجداول العلاقية بها فى الشكل الطبيعى الثانى.

الشكل الطبيعى الثالث Third Normal Form (3NF) :

يكون الجدول العلاقى فى الشكل الطبيعى الثالث (3NF) لو كان فى الشكل الطبيعى الثانى ، وإن كل خاصية لاأولية فى الجدول العلاقى لا تخضع للتبعية الانتقالية على أى مفتاح. بمعنى آخر فإنها لا تتغير إلا مع المفتاح فقط وليس غيره.

مثال (٥-٥) :

افترض أن الفئة الشاملة U تشمل Mang (M) ,Dept (D) , Items (I) , Store (S) والتى يمكن التعبير عنها كالآتى:

$$U = \{S , I , D , M\}$$

وأن التبعيات الوظيفية هى كالآتى:

$$F = \{SI \rightarrow D , SD \rightarrow M\}$$

المطلوب : تحديد الشكل الطبيعى للجدول العلاقى مع وضع الفرضية الدائمة من أن أى جدول علاقى هو فى الشكل الطبيعى الأول.

الحل :

أولاً : حساب الفئات الانغلاقية المختلفة وتحديد أيهما تشمل الفئة الشاملة كالآتى:

$$(SI)^{+} = SIDM = \cup$$

لذا تعتبر SIDM هى المفتاح الشامل وأن SI مفتاح مرشح.

$$(SD)^{+} = SDM \neq \cup$$

$$(DI)^{+} = DI \neq \cup$$

$$(MI)^{+} = MI \neq \cup$$

لذا فإن MI , DI , SD لا تعتبر مفاتيح. أما I , S فهما من الخصائص الأولية ، فى حين أن D , M من الخصائص اللاأولية . وأن SI هو المفتاح الأساسى.

وأن الفئة الانغلاقية للتبعيات الوظيفية هى:

$$F^{+} = \{SI \rightarrow D , SI \rightarrow M , SD \rightarrow M\}$$

ثانيًا : الشكل الطبيعى الثانى : ينتهك فقط إذا كان:

$$S \rightarrow D \mid I \rightarrow D \mid SI \rightarrow M \quad I \rightarrow M$$

لأنه عندئذ تكون D و M خاصيتين لاأوليتين وسيكونان ذَوَى تبعية جزئية على SI. ولكن فى هذه الحالة D و M الخصائص اللاأولية وذات تبعية كاملة ؛ لذا فإن الجدول العلاقى فى الشكل الطبيعى الثانى.

ثالثًا: الشكل الطبيعى الثالث :

على الرغم من أن SD ليس مفتاحًا وأن:

$SI \rightarrow SD$

$SD \rightarrow M$

من هذا يتبين أن الخاصية M ذات تبعية انتقالية على المفتاح ؛ لذا فإن الشكل الطبيعى الثالث انتهك.

مثال (٥-٦) :

افترض أن الفئة الشاملة U تشمل (Item (I) , Addr (A) , Supp (S) , Price (P يمكن التعبير عنها كالآتى:

$$U = \{S , A , I , P\}$$

وأن التبعيات الوظيفية F هى:

$$F^{+} = \{SI \rightarrow SD, \; SD \rightarrow M\}$$

المطلوب : تحديد الشكل الطبيعى للجدول العلاقى مع وضع الفرضية الدائمة من أن العلاقة بالشكل الطبيعى الأول.

الحــل :

أولاً : حساب الفئات الانغلاقية المختلفة وتحديد أيهما تشمل الفئة الشاملة.

$(S)^{+} = SA \neq U$

$(SI)^{+} = SLAP = U$

$(^{+}IP) = IP \neq U$

لذا تعتبر SIAP المفتاح الشامل ، ويعتبر SI المفتاح المرشح.

$$(SP) = SAP \neq U$$

ثانيًا – الشكل الطبيعى الثانى:

ويتبين من التبعيات الوظيفية F أن:

$$SI \rightarrow A$$

$$S \rightarrow A$$

ويتضح أن الخاصية A ذات تبعية جزئية على المفتاح المرشح ، ومن هنا يتبين أن الشكل الطبيعى الثانى قد انتهك.

مثال (٥–٧):

نفرض أن الفئة الشاملة U تشمل City(C) , Street(S), ZipCod(Z) والتى يمكن التعبير عنها كالآتى:

$$U = \{C , S , Z\}$$

وأن التبعيات الوظيفية F هى :

$$F = \{CS \rightarrow Z, \quad Z \rightarrow X\}$$

المطلوب : تحديد الشكل الطبيعى للجدول العلاقى مع وضع الفرضية الدائمة من أن الجدول العلاقى بالشكل الطبيعى الأول.

الحـــل :

أولاً : حساب الفئات الانغلاقية المختلفة وتحديد أيهما تشمل الفئة الشاملة.

$$(CS)^{+} = CSA = \cup$$

$$(Z)^{+} = ZC \neq \cup$$

$$(ZS)^{+} = ZCS = \cup$$

$$(ZS)^{+} = ZC \neq \cup$$

لـــذا فإن CSZ تعتبر المفتاح المرشح وإن CS تمثل المفتاح الأساسى.

ثانيًا – الشكل الطبيعى الثانى :

من التبعات الوظيفية F التالية :

$$CS \rightarrow Z$$

$$Z \rightarrow C$$

يتبين أن الخاصية Z ذات تبعية كاملة على المفتاح الأساسى CS. ومن هذا يتبين أن الجدول العلاقى فى الشكل الطبيعى الثانى.

ثالثًا – الشكل الطبيعى الثالث :

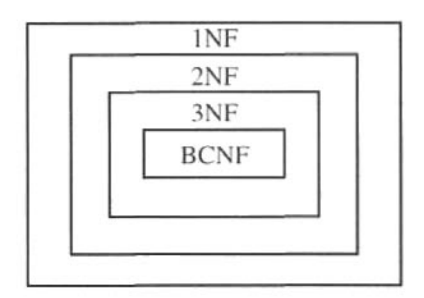
كما هو واضح أن خاصية Z خاصية لاأولية وليس لها تبعية انتقالية على المفتاح الأساسى CS ؛ لـذا فإن العلاقة فى الشكل الطبيعى الثالث.

الشكل الطبيعى بويس – كود (BCNF) :

يعتبر الشكل الطبيعى الثالث كافيًا لمعظم التطبيقات العملية لقواعد البيانات ولكنه لا يضمن إزالة كل الأخطاء. فقد تحدث مشكلة إضافة أو تحديث أو حذف فى حالة وجود أكثر من مفتاح مرشح فى جدول علاقى بالرغم من أنها فى الشكل الطبيعى الثالث.

وتكون العلاقة فى الشكل الطبيعى بويس - كود (BCNF) إذا كانت جميع الخصائص الأولية ذات تبعية كاملة على أى مفتاح. بمعنى آخر أن كل محدد فى الجدول العلاقى يجب أن يكون مفتاحًا. ويلاحظ أنه لو كان الجانب الأيسر من التبعية الوظيفية F ليس مفتاحاً فإن ذلك سيؤدى إلى تبعية جزئية ولا يصبح الجدول العلاقى فى الشكل الطبيعى بويس – كود. ويوضح الشكل رقم (٥–١٠) تسلسل الأشكال الطبيعية.

شكل رقم (٥-١٠) يوضح تسلسل الأشكال الطبيعية



مثـــال (٥-٨):

فى المثال (٥-٧) حيث يوجد الجدول العلاقى فى الشكل الطبيعى الثالث.

المطلوب: تحديد ما إذا كان الجدول العلاقى يخضع للشكل الطبيعى بويس ـ كود أم لا.

الحــل:

حيث إن كلاً من ZS ، CS تشكل مفتاحًا أساسيًا ، وإن التبعيات الوظيفية F:

$$CS \rightarrow Z$$

$$Z \rightarrow C$$

وإن الجانب الأيسر Z من التبعية الوظيفية الثانية ليس مفتاحًا شاملاً أو مفتاحًا مرشحًا فإن الجدول العلاقى ليس فى بويس – كود BCNF.

خطوات تفكيك بدون فقدان ربط فى الشكل الطبيعى بويس – كود :

المدخلات: مخطط الجدول العلاقى R والتبعيات الوظيفية F

المخرجات: تفكيك الجدول العلاقى R بدون فقدان دمج. حيث إن كل مخطط جدول علاقى فى التفكيك يتم إثبات أنه فى بويس – كود ، يرفض من التبعيات الوظيفية F لذلك المخطط.

الطريقــة:

(١) يتكون التفكيك فى البداية من مخطط الجدول العلاقى R فقط.

(٢) إذا كــان S هو مــخطط الجـدول العــلاقى فى التــفكيك ، وثبت أنه ليس فى البويس–كود.

عندئذ يتم جعل التبعية الوظيفية التالية:

$$X \rightarrow A$$

فى S مخطط الجدول العلاقى حيث إن الخاصية X ليست مفتاحًا لذلك المخطط.

(٣) يتم استبدال مخطط الجدول العلاقى S عند تكرار التفكيك بكل من مخططى العلاقيين S2 , S1 ، حيث إن مخطط الجدول العلاقى S1 يتكون من الخاصية A والخـاصيـة X ، S2 تتكون من كل الخصـائص الموجـودة فى مـخطط الجـدول العلاقى S ماعدا الخاصية A وهكذا يكون:

مخططا الجدولين العلاقيين S2 , S1 فئتين جزئيتين مناسبتين من مخطط الجدول العــلاقى S ، وأن X هى مـفـتـاح شـامل له. وأن تقـاطـع كل من مـخططى الجـدولين العلاقيين S2 , S1 يساوى X. أى أن:

وإن الفرق بين مخططى الجدولين العلاقيين S2 , S1 يساوى A. أى أن:

(٤) كل من مخططى الجدولين العلاقيين S1 ، S2 ينبغى أن يحتويا على خصائص أقل مما هو فى مخطط الجدول العلاقى S. وهكذا كل مخطط جدول علاقى له خاصيتين أو أقل يجب أن يكون فى الشكل الطبيعى بويس – كود (BCNF).

مثال (٥–٩):

ضـع مـخطط الجـدول العـلاقى R ذا الخـصـائص الـتـاليـة فـى الشكل الطبـيـعى بويس – كود:

المادة : Course (C) ، المدرس : Teacher (T) ، الوقت المخـصص للمـادة: Hour (H) ، غرفة التدريس : Room (R) ، الطلاب: Student (S) ، التقديرات : Grades (G).

وبالتالى يمكن التعبير عن مخطط الجدول العلاقى R كالآتى :

$R\ (C, T, H, R, S, G)$

القيــود :

(١) كل مادة (C) لها مدرس (T) واحد فقط.

(٢) المادة (C) الواحـــدة فـقط يجب أن تكون فى غـرفـة التــدريس (R) فى الوقت المخصص للمادة (H) .

(٣) مـدرس (T) واحـد فـقط يجب أن يكون فى غـرفـة التــدريس (R) فى الوقت المخصص للمادة (H) .

(٤) كل طالب (S) له تقدير (G) واحد فقط فى كل مادة (C) .

(٥) الطالب (S) يجب أن يكـون فـى غـرفـة التدريـس (R) فـى الوقت المخـصص للمادة (H) .

الحـــل:

الفئة الشاملة هى :

$U = \{ C, T, H, R, S, G \}$

التبعيات الوظيفية المستنتجة من القيود هى:

$F = \{ C \rightarrow T,\ HR \rightarrow C,$

$HT \rightarrow R, CS \rightarrow G, HS \rightarrow R \}$

حساب الفئات الانغلاقية:

$(C)^+ = CT \neq \cup$

$(HR)^+ = HRCT \neq \cup$

$(HS)^+ = HSRCTG = \cup$

بناءً على **خطوات التفكيك** فإنه يتم اختيار أى فئة انغلاقية عدا $(HS)^+$ التى تمثل المفتاح المرشح. ومن هنا يتبين أن مخطط الجدول العلاقى R ليس فى الشكل الطبيعى بويس – كود؛ لأنه ليس كل الجوانب اليسرى فى التبعيات الوظيفية F تمثل مفتاحاً. ولتفكيك هذ الجدول العلاقى إلى الشكل الطبيعى بويس – كود ، نأخذ التبعية الوظيفية GS ‹ G حيث إنها تنتهك شرط الشكل الطبيعى بويس – كود لأن CS ليس مفتاحاً.

(١) تفكيك مخطط الجدول العلاقى الأصلى R (C, T, H, R, S, G) إلى كل من المخططين R1 (C, S, G) , R2 (C, T, H, R, S). حيث إنه يمكن اختيار التبعية الوظيفية.

(٢) إنه من السهل إثبات أن مخطط العلاقة R1 فى الشكل الطبيعى بويس – كود.

(٣) يجب تفكيك مخطط الجدول العلاقى R2 (C, T, H, R, S) وتختار التبعية :

$$C \rightarrow T$$

لتقسيم مخطط الجدول العلاقى R2 إلى مخططين ، هما: R4 (C, H, R, S) , R3 (C, T).

ومن الواضح أن مخطط الجدول العلاقى R3 يخضع للشكل الطبيعى بويس – كود. ومن ثم ينبغى إعادة تقسيم مخطط الجدول العلاقى R4 باستخدام التبعية الوظيفية:

$$CH \rightarrow R$$

وفى النهاية سوف يتم الحصول على المخططات التالية للجدول العلاقى R (C, T, H, R, S, G) وهى:

- R1 (C, S, G)
- R3 (C, T)
- R5 (C, H, R)
- R6 (C, H, S)

حيث يلاحظ أن التبعية الوظيفية :

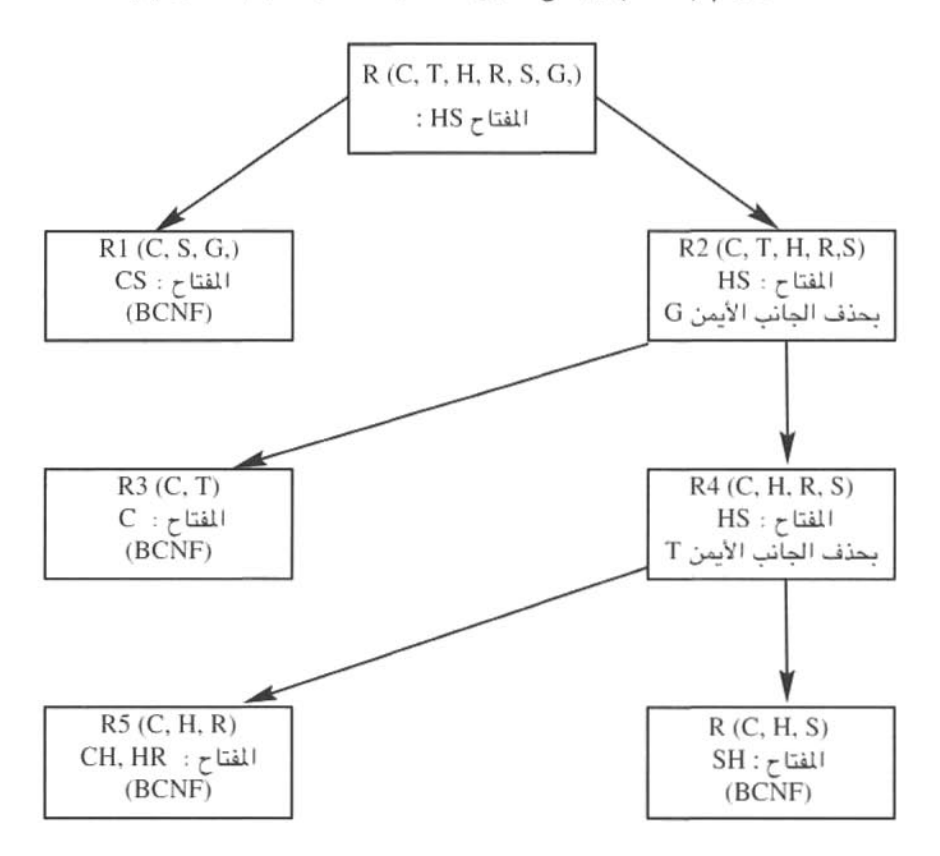
$$CH \rightarrow R$$

تؤدى إلى تغطية المخطط العلاقى R4 حيث يتبعها التبعيات الوظيفية التالية:

$$C \rightarrow R$$

$$HT \rightarrow R$$

الشكل رقم (٥-١١) يوضح شجرة التفكيك حسب التبعيات الوظيفية



عيوب التفكيك :

(١) أنه لا يحتفظ ببعض التبعيات الوظيفية كما فى المثال رقم (٥-٩) حيث تم فقد هذه التبعية:

$$HT \rightarrow R$$

(٢) ليس هناك مؤشر يبين أى الطرق أسرع فى الوصول إلى مخططات الجداول العلاقية للشكل الطبيعى بويس - كود. وإنما تعتمد على خبرة المصمم وكذلك سهولة التصميم .

الشكل الطبيعى الرابع Fourth Normal Form(4NF) :

قد تحدث مشكلات فى حالات الإضافة أو الحذف أو التحديث فى حالة وجود تبعيات متعددة القيم (Multivalued) بين الخصائص فى جدول علاقى معين بالرغم من أنه فى الشكل الطبيعى بويس - كود: ولذا يكون الجدول العلاقى فى الشكل الطبيعى الرابع إذا كان فى الشكل الطبيعى بويس - كود BCNF ولا تحتوى على قيم متعددة [١]،[٣].

ويتبين ذلك من دراسة الجدول العلاقى الخاص "بالفرص الدراسية" OFFERING كما فى الشكل رقم (٥-١٢). وأن القواعد الدلالية لهذه العلاقة كالآتى:

(١) لكل مادة (Course) يوجد أكثر من محاضر (Instructor) وأكثر من مرجع (Textboob).

(٢) المحاضرون والمراجع تابعون مرتبطون بالمادة فقط فى حين أنهم هم مستقلون عن بعضهم.

شكل رقم (٥-١٢) الجدول العلاقى الخاص "بالفرص الدراسية"

OFFERING

Course	Instructor	Text-Book
Management	Ali	Drucker
Management	Ahmed	Drucker
Management	Saad	Drucker
Management	Ali	Peters
Management	Ahimed	Peters
Management	Saad	Peters
Finance	Gamil	Weston
Finance	Gamil	Gulford

بفحص الجدول العلاقى الخاص "بالفرص الدراسية" OFFERING فى الشكل رقم (٥-١٢) نجد أنه فى الشكل الطبيعى بويس - كود ، ولكن يوجد شكل من أشكال التبعية بين الخصائص. وتكمن هذه التبعية فى أنه لكل مادة مجموعة من المحاضرين ومجموعة من المراجع. وتسمى هاتان التبعيتان بـ "التبعيات متعددة القيم" ويرمز لها كالآتى:

COURSE →→ INSTRUCTORS

COURSE →→ TEXTBOOK

وتحدث هذه التبعيات عندما توجد ثلاث خصائص (A, B, C) فى الجدول العلاقى، ولكل قيمة (A) توجد مجموعة قيم (B) ومجموعة قيم (C) ولكن قيم (B) مستقلة عن (C) والعكس بالعكس. ولإزالة هذه الأخطاء من الجدول العلاقى الخاص "بالفرص الدراسية" OFFERING ينبغى تقسيمه إلى جدولين علاقيين للتخلص من التبعيات متعددة القيم. وحينئذ تصبح الجداول العلاقية الناتجة فى الشكل الطبيعى الرابع كما فى الشكل رقم (٥-١٣).

الشكل رقم (٥-١٣) الجداول العلاقية الناتجة عن تقسيم الجدول العلاقى الخاص "بالفرص الدراسية" OFFERING فى الشكل الطبيعى الرابع

TEACHER

Course	Instractor
Management	Ali
Management	Ahmed
Management	Saad
Finance	Gamil

TEXT

Course	Textbook
Management	Drucker
Management	Peters
Finance	Weston
Finance	Gulford

الشكل الطبيعى الخامس Fifth Normal Form (5NF) :

تكون العلاقة فى الشكل الطبيعى الخامس (5NF) إذا كانت فى الشكل الطبيعى الرابع وليس بها تبعيات الربط، والتى تبين إمكانية إعادة إنشاء الجدول العلاقى الأصلى من الجداول العلاقية الناتجة من التقسيم دون الحصول على مجموعات للقيم المرتبة (صفوف) زائدة أو ناقصة. ويكون الجدول العلاقى به تبعية ربط فى حالة عدم القدرة على إعادة بناء الجدول العلاقى الأصلى من الجداول العلاقية الناتجة عن التقسيم[١].

ويجب أن يلاحظ أن تبعيات الربط نادرة الحدوث وأن الشكل الطبيعى الثالث كان فى معظم الأحوال أنسب للتطبيقات العملية.

الطريقة الاصطناعية للتصميم Synthetical approach :

وتضمن هذه الطريقة فى تصميم قاعدة البيانات الخطوات التالية [٥، ٢]:

١- التعامل مع الفئة الشاملة U والتبعيات الوظيفية F للحصول على قاعدة البيانات.

٢- قراءة الحروف حسب ترتيب تكرارها.

٣- تعتمد على الرسومات Graphs والمنطقيات Logics وتفترض أن التبعيات لها متكافئة فى منطقيتها.

تمثيل الرسم لمجموعة تبعيات :

يتكون الرسم Graph من الرؤوس Vertices والوصلات Edges ويعبر عنه كالتالى:

$$G = (V , E)$$

وتتكون الرؤوس (V) من مجموعتين ، يمكن أن يرمز لإحداهما Vs والأخرى V^C. حيث تمثل V^s مجموعة الرؤوس التى تشمل الخصائص الفردية فى الجانب الأيسر من التبعيات الوظيفية مثل (A > D) فى حين تمثل V^C مجموعة الرؤوس للخصائص المركبة فى الجانب الأيسر من التبعيات الوظيفية مثل (AB > C) ويعتبر عن مجموعة الرؤوس V كالآتى:

$$V = \Sigma Vs + Vc$$

وتتكون الوصلات (E) من مجموعتين ، يمكن أن يرمز لإحداهما بالرمز E^F والتى تمثل الوصلة الكاملة بين رأسين فى حين تمثل E^d الوصلة المنقطة التى تصل بين الرأس الممثل للخصائص المركبة وبين محتويات هذه الخصائص الأصلية.

مثال (٥-١٠) :

بفرض أن الفئة الشاملة لقاعدة البيانات هى:

$$U = (A, B, C, D, E, F, G)$$

وبناء على الطريقة الاصطناعية فى التصميم فإن:

$$G = 7 , F = 6 , E = 5 , D = 4 , C = 3 , B = 2 , A = 1$$

وهكذا كل خاصية فى الفئة الشاملة تأخذ رقمًا حسب ترتيبها. وبفرض أن التبعيات الوظيفية هى:

$$F = \{A \rightarrow BCF , C \rightarrow D, BD \rightarrow E, EF \rightarrow G\}$$

وهكذا فى التبعيات الوظيفية يجب أن يعطى الجانب الأيسر للخصائص المركبة رقمًا متتاليًا فى القيمة لآخر رقم تم الوصول إليه. ومن ثم فإن:

$EF = 9$, $BD = 8$

لـذا فإن فئة الرؤوس V تكون كالآتى :

$V= \{ 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 \}$

وتكون فـئـة الوصـلات الكاملة E^F بناء على القيم الرقـمـيـة للخـصـائص الفـردية والخصائص المركبة كالآتى:

$$E^F = \{ (1,2) , (1,3) , (1,6) , (3,4) , (8,5) , (9,7) \}$$

فحين تكون فئة الوصلات المنقطة E^d كالآتى:

$$E^d = \{ (9,5) , (9,6) , (8,2) , (8,4) \}$$

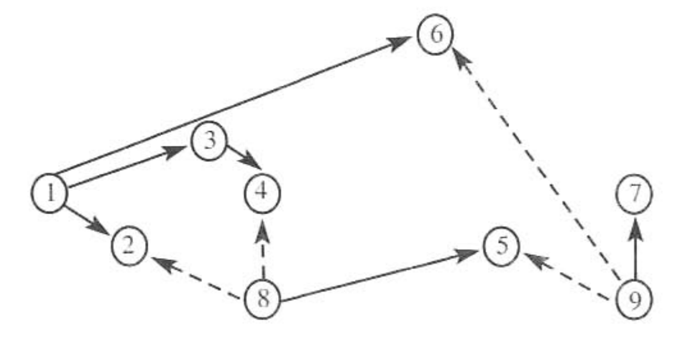
المطلوب : استنتاج الرسم التبعى Dependency Graph .

الحــل:

أولاً : يتم تمثيل فئة الرؤوس V .

ثانيًا: تستكمل الوصلات الكاملة والمنقطة بناءً على الفئتين E^F ، E^d والتى يتم تشكيلهما حسب التبعيات الوظيفية وبناء عليه يتم الحصول على الرسم كما بالشكل رقم (٥–١٤).

شكل رقم (٥ – ١٤) يبين تمثيل الرسم للتبعيات الوظيفية



المشكلة : فى الرسم السابق هى القدرة على تحديد المسار بين رأسين.

الرسم الضمنى Implication Graph (IG) :

هو رسم فرعى Subgraph من الرسم التبعى، يتم تمثيله مباشرة ويرمز له بالرمز G (v.e) : حيث إن أى رأس يحتويها لابد أن تنتمى إلى فئة الرؤوس. أى أن :

$$v \in V$$

وكذلك أى وصلة لا بد أن تنتمى إلى فئة الوصلات. أى أن:

$$e \in E$$

وهذا الرسم الضمنى يتم تعريفه بين رأسين فقط. وهكذا فى حالة تواجد رسم فرعى من الرسم الأصلى G يحقق شروط الضمنية ، عندئذ يوجد ضمنيًا بين رأسين.

بصفة عامة كما تم توضيحه من قبل ، فإن مشاكل تصميم قاعدة البيانات تبزغ من البيانات الزائدة عن الحد حتى فى الطريقة التحليلية للتصميم. ففى أى شكل طبيعى عندما تكون البيانات زائدة عن الحد فإن التصميم النهائى سوف يتضمن هذه الزيادة. ولكن لتغطية هذه الزيادات فى الطريقة التحليلية ، كان لابد اضطراريًا تحديد الفئة الانغلاقية للتبعيات الوظيفية E^F. بينما لا تعتمد الطريقة الاصطناعية على تغطية هذه الزيادات باستخدام الفئة الانغلاقية للتبعيات الوظيفية E^F .

تمثيل الرسم الضمنى :

لتمثيل الرسم لابد من توافر التبعيات الوظيفية للرسم G (V . E) والرؤوس المميزة له (مثل i . j) فى الفئة V. والرسم الضمنى من الرأس الأول i إلى الرأس الأخير j هو رسم فرعى من الرسم الأصلى. بمعنى آخر نجد أن:

$$G(i, j) = (V_{ij}, E_{ij})$$

لمجموعة التبعيات الوظيفية للرسم. وفى حالة وجود رسمى ضمنى بين رأسين ، فإن الجدول العلاقى بين هذين الرأسين تكون زائدة عن الحد ولا بد من إزالتها. وتبدأ القواعد التالية (الحالات) بمساواة x بالرأس j وتعيد تعريف نفسها بإنشاء رسم فرعى يعتمد على صفات الرأس النهائى x.

قاعدة (الحالة) ١:

يكون الرأس النهائى x للرسم G (i , x) بسيطًا. وعندما يوجد الرأس k ، حيث إن الوصلة الكاملة (k . x) تكون متضمنة فى الوصلة E_{ix} .

K ⟶ j (x)

(١) الوصلة (i.k) تنتمى إلى الوصلة Eix. أى أن كلاً من الوصلة الكاملة أو المنقطة تكون بين i, k.

i - - - - - -➤ K ⟶ j

حيث إن j تمثل رأسًا بسيطًا والوصلة (k,j) تكون وصلة كاملة.

(٢) فى الرسم الضمنى توجد الوصلة من الرأس i إلى الرأس k وتنتمى إلى G (i,x) عندما لا تنتمى الوصلة (i.k) إلى E.

i ⬭ K ——— j (x)

القاعدة (الحالة) ٢:

الرأس النهائى x للرسم G (i,x) تكون مركبة وذات رؤوس مكونة لها ، ويمكن أن يرمز لها كالتالى حيث إنه يفترض أن عدد هذه الرؤوس (r) :

m1 , m2 , ... , mr-1 , m

وتنتمى الوصلات المنقطة التالية:

(x,m1) , (x,m2), ... , (x,mr)

إلى Eix . ويكون رأس واحد على الأكثر من عدد الرؤوس (r) مساويًا للرأس i ولكل محتوى رأس يتبين أن m_s لا تساوى i حيث إن s تنتمى إلى الفئة {r , ... ,٣ ,٢ ,١} ولابد أن تكون الوصلة الخارجة من i وصلة كاملة.

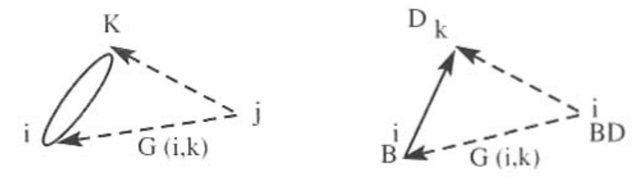
(١) الوصلة (i,m_s) توجد فى Eix سواء أكانت وصلات كاملة أم منقطة حيث إن:

$$i, m_s \in E$$

(٢) الرسم الضمنى من i إلى m_s يوجد ويكون فى الرسم G (i,x) عندما (i,ms) لا تنتمى إلى الوصلة E.

الحالـة الأولى:

شكل رقم (٥-١٥) يوضح الرسم الضمنى G (i,k)



ولابد من التـأكيـد على أن الوصلة الخـارجـة من الرأس i تكون وصلة كـاملة ، ويستحيل أن تكون منقطة: لأن الرأس i يجب أن يكون بسيطًا لأنه أحد مكونات الرأس المركب j .

الحالة الثانيـة:

فى المثال رقم (٥-١٠) الذى يتضمن الفئة الشاملة U التالية:

$$U = (A, B, C, D, E, F, G)$$

والتبعيات الوظيفية F هى:

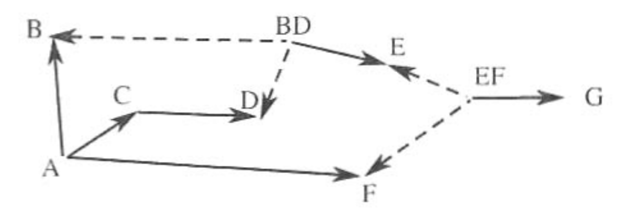
$$F = \{A \rightarrow BCF\ ,\ C \rightarrow D, BD \rightarrow E, EF \rightarrow G\}$$

والسؤال: هل يوجد رسم ضمنى من الرأس A إلى الرأس G ؟.

الحـل: يوجـد رسم ضـمنى عندمـا يخـرج من الرأس A على الأقل وصلة كـاملة واحـدة. ويتم توضيـح الحل برسم الرؤوس والوصـلات الكاملة والمنقطة. وفى حالة عدم

وجود وصلة كاملة لا يمكن أن يتم الحصول على رسم ضمنى. ويبين الشكل رقم (٥-١٦) الرسم الضمنى.

شكل رقم (٥-١٦) يوضح الرسم الضمنى G (A , EF) للرسم G (A,G)



والذى يمكن تبسيطه كما فى الشكل رقم (٥-١٧).

شكل رقم (٥-١٧) تبسيط للرسم الضمنى G (A,G)

A EF G

مثال (٥-١١):

تبين الفئة الشاملة U لقاعدة البيانات الخصائص التالية:

$U = (A, B, C, D, E, F)$

والتبعيات الوظيفية الآتية :

$F = \{AB \rightarrow E.CD \rightarrow F.A \rightarrow C.B \rightarrow C,B \rightarrow D.C \rightarrow A \rightarrow D \rightarrow B.F \rightarrow AD\}$

المطلوب: استخدام الرسم التبعى فى تحديد الرسم الضمنى

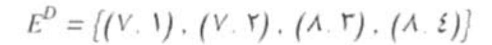
الحـل: الرسم التبعى G (V,E) حيث إن:

$V = \{1, 2, 3, 4, 5, 6, 7, 8, \}$

وكلٌ من الخصائص المركبة بالجانب الأيسر تأخذ رقمًا كما هو معروف على سبيل المثال: AB = 7 ، CD = 8 . وأن عدد الوصلات الكاملة يساوى عدد الخصائص الموجودة بالجانب الأيمن للتبعيات الوظيفية؛ لـذا فإن:

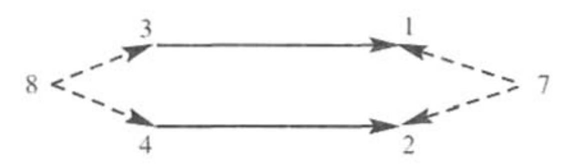
$$E^F / (V.٥) = (٨.٦) , (١.٣) , (٢.٤) , (٣.١) , (٤.٢) , (٦.١) , (٦.٤)\}$$

بينما يكون عدد الوصلات المنقطة مساوياً لعدد المحتويات المكونة للرؤوس المركبة ؛ ومن ثم فإن :



$$E^D = \{(V.١) , (V.٢) , (٨.٣) , (٨.٤)\}$$

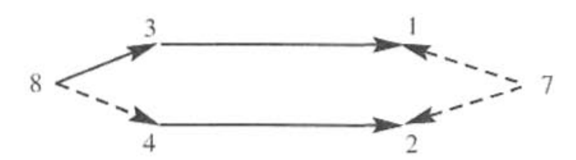
شكل رقم (٥–١٨) الرسم الضمنى المنقوط



وكما هو معروف أنه إن لم توجد وصلة كاملة على الأقل بالرسم لا يمكن الحصول على رسم ضمنى ويسمى هذا الرسم بالرسم الضمنى المنقوط؛ لأن جميع الوصلات الخارجة من الرأس 8 لا تحتوى على وصلة كاملة.

ومن جانب آخر: لو فرض أن الرسم كان كما بالشكل رقم (٥–١٩).

شكل رقم (٥–١٩) الرسم الضمنى الكامل



ونظراً لأن الرأس 8 خرج منه وصلة كاملة (على الأقل) فإنه يسمى الرسم الضمنى الكامل .

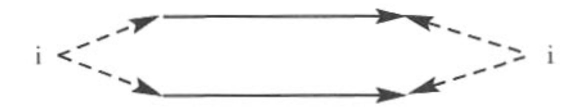
الرأس الزائد عن الحد :

الرأس المركب i فى الرسم التبعى يكون زائداً لو أن كل وصلة من الرأس i إلى الرأس j فى الوصلات E زائدة. بمعنى آخر فإنه عندما لا توجد وصلة خارجة أو داخلة للرأس i، وحذفها لا يغير التبعيات الانغلاقية.

الوصلة الزائدة عن الحد :

(١) لأى وصلة كاملة بين رأسين j . i ، ويوجد رسم ضمنى منقوط بين هذين الرأسين ، عندئذ تكون الوصلة زائدة كما هو موضح بالشكل رقم (٥-٢٠). ودائمًا تكون الوصلات الزائدة وصلات كاملة وليست منقطة.

الشكل رقم (٥-٢٠) يبين الوصلات الزائدة عن الحد



(٢) عندما تكون الوصلة (i . j) فى الوصلة E^f والرسم الضمنى (الكامل أو المنقوط) لا يحتوى على وصلة كاملة من i إلى j فإن الوصلة (i. j) تكون زائدة.

مقدمة فى لغة الاستعلام البنائيـة :

قد تم تطبيق هذه اللغة على قواعد البيانات العلاقية ، وهى لغة بيانات فرعية Data Sub-Language لأنها لا تحتوى على تعليمات للتحكم بالإضافة إلى سهولة تعلمها. وأصبحت لغة أكثر أهمية بعد أن تم استعمالها حاليًا لنظم قواعد البيانات العلاقية لشركة أى .بى . إم IBM لكل من .DB2 , SQL / DS وتعتبر أوراكل ORACLE هى أحد نظم قواعد البيانات العلاقية الأخرى التى تستعمل لغة الاستعلام البنائية SQL والتى تلقى انتشارًا واسعًا فى الحاسبات الإلكترونية بأنواعها المختلفة (كبيرة الحجم - متوسطة الحجم - صغيرة الحجم). وقد قام معهد المقاييس القومية الأمريكية ANSI بإضفاء الصفة الرسمية لها بوصفها لغة قياسية لقواعد البيانات العلاقية. ويرجع السبب وراء ذلك إلى أن التركيب القياسى المقترح كان موجهًا نحو نسخة لغة استعلام بنائية مضمنة embedded فى أى لغة برمجة أكثر منها لغة استعلام بنائية بمفردها. وحاليًا تدعم معظم إدارة قواعد البيانات العلاقية المعالجة المباشرة Interactive SQL ومضمنة فى برامج لغات الجيل الثالث Embeded SQL.

مميزات لغة الاستعلام البنائية SQL :

١- لغة علاقية قياسية مشابهة للغة الانجليزية.

٢- لغة بسيطة ومرنة وقوية فى تداول بيانات الجداول ، فهى لغة :

* لاإجرائية Non Procedural

* تسمح بمعالجة السجلات على مستوى "مجموعة السجلات".

* تسمح بالمعالجة المباشرة Interactive SQL

يتمكن كل مستخدم للغة الاستعلام البنائية SQL من تداول ومعالجة مجموعة من الجداول التى تناظر قيوداً مختلفة، تشكل وحدة كاملة من هذه الجداول وصلاحيات تداول البيانات منظور المستخدم لقاعدة البيانات. وفى أى حالة ، بمقدور المستخدم أن يحصل على التداول التام لكل الجداول تم إنشاؤها.

وتتكون لغة الاستعلام البنائية SQL من تعليمات لتعريف البيانات ولمعالجتها وللتحكم فيها. ويتم استخدام لغة تعريف البيانات (Data Definition Language (DDL فى إنشاء وحذف وتعديل جداول قاعدة البيانات. ويتم استخدام لغة معالجة البيانات (Data manipulation Language (DML فى استرجاع وصيانة (إضافة وحذف وتعديل) بيانات قاعدة البيانات. أما لغة التحكم فى البيانات (Data Control Language (DCL فيتم استخدامها فى تحديد صلاحيات التداول. وسوف يتم طرح فكرة مبسطة عن كيفية إنشاء ومعالجة مخططات قاعدة البيانات باستخدام لغة الاستعلام البنائية SQL [٢].

أنواع بيانات لغة الاستعلام البنائية SQL :

تعتمد أنواع البيانات على نظام إدارة قواعد البيانات المستخدم ومنها[٥]:

١- بيانات رقمية:

* أرقام صحيحة (موجبة أو سالبة). تتراوح فى المدى (٣٢.٧٦٧- ٣٢.٧٦٨) وتعرف بكلمة SmallInt .

* أرقام صحيحة (موجبه وسالبه) . تتراوح فى المدى (٢,١٤٧,٤٨٣,٦٤٧- ٢,١٤٧,٤٨٣,٦٤٨) ويتم تعريفها بكلمة Integer (INT) .

* الأرقام الحقيقية. يتكون الرقم من عدد "m" من الخانات. منها "n" على يمين الفصلة العشرية. ويتم تعريفها بكلمة NUMBER (m,n) .

٢- بيانات حرفية :

* سلاسل حرفية ثابتة الطول. المدى "n" (١-٢٥٥) حرفًا. ويتم تعريفها بكلمة CHAR(n).

* سلاسل حرفية متغيرة الطول. المدى "n" (١-٢٠٠٠) حرفًا.

٣- بيانات الوقت والتاريخ :

* بيانات تاريخ DATE.

* بيانات وقت TIME.

* بيانات وقت وتاريخ TIMESTAMP.

أولاً: تعريف البيانات Data Definition(DDL) [٥)،(٨] :

أمر إنشاء جدول CREATE Table :

المستخدمون قد يغيرون مخططاتهم Schemas بواسطة أوامر معالجة المخطط المختلفة. أمر الإنشاء CREATE Statement يستعمل لإضافة جدول إلى المخطط .

*** التركيبة اللغوية لأمر إنشاء جدول Create table :**

```
CREATE TABLE < table name >
( < Calumn definition > [{,< Colamn definition > } ...] :
```

* **التركيبة اللغوية لعبارة تعريف العمـود Column definition:**

< Column name > < data type > [Not NULL]

يتم توصيف العمود المناظر لأى خاصية. وقد لا يسمح لعمود معين بألا يحتوى على قيمة خالية NULL باستعمال عبارة "ليست قيمة خالية" .NOT NULL تختلف القيمة الخالية عن المسافة blank أو الصفر، وهى تشير إلى قيمة معينة قد تكون غير معروفة أو قيمة متاحة. على سبيل المثال: "عنوان موظف" قد يكون غير معروف فى وقت معين، وفى هذه الحالة يمكن استخدام القيمة الحالية NULL.

مثـال (٥-١٢):

يبين هذا المثال كيفية إنشاء مخططات قاعدة بيانات "الإدارة" كما فى الشكل رقم (٥-٢١) وواقعة كل مخطط كما فى الشكل رقم (٥-٢٢).

أ- مخططات قاعدة البيانات :

شكل رقم (٥-٢١) مخططات قاعدة بيانات الإدارة

* إنشاء جدول الإدارة

```
* CREATE TABLE DEPARTMENT
  (DNAME CHAR (15) NOT NULL ,
  LOCATION CHAR (2) ,
  MANAGER CHAR (15) ) ;
```

* إنشاء جدول المهام

```
* CREATE TABLE TASK
  (DNAME CHAR (15) NOT NULL ,
  TASKNO INTEGER NOT NULL ,
  TASKNAME CHAR (15)) ;
```

* إنشاء جدول الموظفين

```
* CREATE TABLE EMPLOYEE
  (DNAME CHAR (15) NOT NULL ,
  SSN CHAR (9) ,
  NAME CHAR (15) ,
  SALARY INEGER ,
  ADDRESS CHAR (20)) ;
```

* إنشاء جدول الحالة

```
* CREATE TABLE STATUS
  (DNAME CHAR (15) NOT NULL ,
  SSN CHAR (9) ,
  TASKNO INTEGER ,
  ST CHAR (10) ) ;
```

ب – وقائع جداول قاعدة البيانات

شكل رقم (٥–٢٢) وقائع جداول قاعدة بيانات «الإدارة»

* واقعة جدول الإدارة DEPARTMENT :

DNAME	LOCATION	MANAGER
Engineering	12	Al-SALEH
Sales	05	JAFER
Training	03	AL-ASKAR
Programs	10	AL-HAJAS
Library	07	AL-ARFAJ

* واقعة جدول المهام TASK :

DNAME	TASKNO	TASKNAME
Engineering	1	Netwprk
Engineering	2	Math
Training	3	Registration
Sales	4	Big sale
Programs	5	Schedules
Library	6	Scripts
Library	7	Books

شكل رقم (٥-٢٢) وقائع جداول قاعدة بيانات "الإدارة"

* واقعة جدول الموظفين EMPLOYEE :

DNAME	SSN	NAME	SALARY	ADDRESS
Engineering	1111	Mansour	25000	17 El Minaa St.
Engineering	2222	Salah	30000	25 Soud King St.
Sales	3333	Mostafa	27000	13 Khalid ring St.
Sales	4444	Sulaiman	26000	02 Al-Maxroaia St
Library	5555	Ahmed	35000	18 Abdel Aziz King St
Programs	6666	Abdel rim	18000	22 Rabie St.
Programs	7777	Ali	2000	32 Najed St.
Traning	8888	Saad	23000	19 Al-Awal St.

* واقعة جدول الحالة STATUS :

DNAME	SSN	Task No
Engineering	1111	1
Engineering	2222	2
Library	5555	7
Programs	6666	3
Sales	3333	4
Library	5555	6
Training	8888	5
Sales	3333	4

*** أمر حذف جدول DROP TABLE :**

أمر الحذف DROP Statement يتم استعماله لحذف جدول من مخطط قاعدة البيانات.

*** التركيبة اللغوية لأمر حذف جدول Drop Table**

DROP TABLE < table name >

مثال (٥-١٣) :

يمكن حذف جدول "الحالة" STATUS من مخطط قاعدة بيانات "الإدارة" كالآتى :-

DROP TABLE STATUS ;

أمر تغيير جدول ALTER :

يستخدم أمر تغيير جدول ALTER Statement لتغيير توصيف الجدول. وذلك بإضافة عمود جديد أو تغيير نوع بيانات العمود (أى بزيادة طول الحد الاقصى للخانة فى الجدول).

*** التركيبة اللغوية لأمر تغيير جدول ALTER TABLE**

* ALTER TABLE < table name > { ADD | MODIFY }

< Alter table clause > ;

*** التركيبة اللغوية لعبارة تغيير جدول alter table clause**

{ < Column name > < data type > } |

{[< Column name > < data type >

[{ , < Column name > < data typ > } ...])}

مثال (٥-١٤) :

يبين المثال التالى إضافة عمود الاسم Name فى جدول "الحالة" STATUS (على افتراض أنه لم يتم حذفه كما تم بالمثال (٥-١٣)).

```
ALTER TABLE STATUS
ADD NAME CHAR (15) ;
```

ثانياً : معالجة البيانات Data Manipulation (DML) [٥]،[٨] :

أمر اختر SELECT Statement :

من أكثر الأوامر أهمية فى لغة الاستعلام البنائية SQL هو أمر "اختر" SELECT Statement. ويعد أمر "اختر" أمراً معقداً، وله اختيارات مختلفة تشتمل على إمكانيات خاصة لثلاث عمليات أساسية فى الجبر العلاقى، هى:

- الربط Join.
- الاختيار الرأس Project.
- الاختيار الأفقى Select.
- التركيبة اللغوية لأمر "أختر" SELECT Statement

على الرغم من تعقيد تركيبة الأمر إلا أنه يمكن صياغتها بشكل مبسط كالآتى:

```
SELECT [ DISTINCT] < Calumn name >
[{ < , name column > } ... ]
< Table expression >
```

* التركيبة اللغوية لعبارة تعبير جدول table expression

```
< FROM > < Table name > [{, < Table name >} ...]
[ < WHERE > < Search condition > ]
[ < GROUP BY Clause > [ < HAVING Clause > ] ]
[ < ORDER  BY Clause ] ;
```

* التركيبة اللغوية لعبارة "جمع حسب" GROUP BY

```
<GROUP BY [ Table name] < Column name >
```

* التركيبة اللغوية لعبارة "يملك" HAVING

```
HAVING < Search Condition >
```

* التركيبة اللغوية لعبارة "رتب حسب" ORDER BY

```
ORDER BY > Column name < [ { . < Column name > } . . . ]
```

يطبع أمر "اختر" SELECT Statement دائمًا النتيجة فى جدول. وسيتم استخدام وقائع قاعدة بيانات "الإدارة" لاستنتاج كل الاستفسارات فى الأمثلة التالية. وتسهيلاً لإيضاح كل حالة (فى معظم الحالات) فإن:

* الأمر يبين فى الجزء (i) .

* والنتيجة يشار إليها فى الجزء (ii).

(أ) الاختيار الرأسى Project

مثــال (٥-١٥):

بفرض أن هناك رغبة فى الحصول على كل أسماء الإدارات فى جدول "الإدارة" DEPARTMENT. وفى هذه الحالة يتم كتابة جميع الخصائص التى نريد اختيارها رأسيًا ، والتى تقتصر فى هذا الاستفسار على خاصية واحدة فقط هى اسم الإدارة Dname. وهذه القائمة متبوعة بعبارة "من" FROM ، يليها أسماء الجداول التى تستدعى منها البيانات. وفى هذه الحالة تتعامل مع جدول "الإدارة" فقط . وسوف يتم توضيح ذلك فيما يلى:

أ- اطبع أسماء الإدارات Dname فى جدول "الإدارة" .

```
(i) SELECT DNAME
    FROM DEPARTMENT ;
```

(ii)

Dname
Engineering
Sales
Training
Programs
Library

ب - اطبع الاسم Name والمرتب Salary لكل موظف .

```
(i) SELECT  NAME , SALARY
    FROM EMPLOYEE ;
```

(ii)

Name	Salary
Mansour	25000
Salah	30000
Mostafa	27000
Sulaiman	26000
Ahmed	35000
Abdel Krim	18000
Ali	20000
Saad	20000

جـ- اطبع كل البيانات لكل موظف يحصل على مرتب أكثر من ٢٥,٠٠٠ ريال سعودى.

```
(i) SELECT DNAME , SSN , NAME , SALARY , ADDRESS
    FROM  EMPLOYEE
    WHERS  SALARY > 25.000
```

(ii)

Dname	SSN	Name	Salary	Name
Engineering	2222	Salah	30000	25 Soud King St.
Sales	3333	Mostafa	27000	13 Khalid ring St.
Sales	4444	Sulaiman	26000	02 Al-Mazroaia St.
Library	5555	Ahmed	35000	18 Abdel Aziz King St.

د- اطبع اسم الادارة Dname واسم الموظف Name والمرتب Salary لكل موظف يحصل على مرتب أكثر من ٢٥,٠٠٠ ريال. وهذا يبين ضم الاختيار الرأسى فى الاستفسار.

(i) SELECT DNAME , NAME , SALARY
FROM EMPLOYEE
WHERE SALARY > 25.000

(ii)

Dname	Name	Salary
Engineering	Salah	30000
Sales	Mostafa	27000
Sales	Sulaiman	26000
Library	Ahmed	35000

هـ- اطبع أسماء الإدارات التى بها موظفون يحصلون على مرتب أكثر من ٢٥,٠٠٠ ريال . بافتراض أننا نركز على أسماء الإدارات فقط.

(i) SELECT DNAME
FROM EMPLOYEE
WHERE SALARY > 25,000

(ii)

Dname
Engineering
Sales
Sales
Library

و- اطبع بدون تكرار كل أسماء الإدارات التى بها موظفون يحصلون على مرتب أكثر من ٢٥.٠٠٠ ريال. ويبين المثال التالى كيفية تجنب التكرار فى الاستعلام .

```
(i) SELECT DISTINCT DNAME
    FROM EMP: OYEE
    WHERE SALARY > 25.000
```

(ii)

Dname
Engineering
Sales
Library

ى- اطبع اسم الإدارة Dname واسم الموظف Name والمرتب Salary لكل موظف يحصل على مرتب أكثر من ٢٥.٠٠٠ ريال. بحيث يتم ترتيب الناتج تصاعدياً حسب المرتب Salary. حيث تستخدم عبارة "رتب حسب" ORDER BY لعمل الترتيب. وفى حالة الترتيب التنازلى تستخدم عبارة "تنازلى" DESC بعد عبارة "رتب حسب" ORDER BY .

```
(i) SELECT DNAME , NAME , SALARY
    FROM EMPLOYEE
    WHERE SALARY > 25.000
    ORDER BY SALARY ;
```

(ii)

Dname	Name	Salary
Sales	Sulaiman	26000
Sales	Mostafa	27000
Engineering	Salah	30000
Library	Ahmed	35000

ل- اطبع الاسم والمرتب الجديد لكل موظف فى إدارة المبيعات بعد زيادة مرتبه بما يعادل ١٠٠٠ ريال.

```
(i) SELECT NAME , SALARY + 1000
    FROM EMPLOYEE
    WHERE DNAME = "SALES" ;
```

(ii)

Name	**Salary + 1000**
Sulaiman	27000
Mostafa	28000

العوامل العلاقية Relational Operators :

الشرط الذى يلى عبارة "حيث إن" WHERE قد يتضمن بعض العوامل العلاقية التالية:

العامل العلاقى	الرمز	العامل العلاقى	الرمز
أكبر من	>	أصغر من أو يساوى	<=
أصغر من	<	يساوى	=
أصغر من	>=	يساوى	<>

العوامل المنطقية Boolean Operators :

إضافة إلى العوامل العلاقية قد تتضمن عبارة "حيث إن" WHERE بعض العوامل المنطقية التالية:

العامل المنطقى	الرمز
تقاطع شرطين	AND
اتحاد شرطين	OR
نفى الشرط	NOT

(ب) الربط Join :

يستخدم أيضًا أمر "اختر" Select للتعبير عن الربط ، إلى جانب استخدامه الموضح فى الأمثلة السابقة فى تنفيذ عمليتى الاختيار الأفقى والرأسى. ويمكن استخدام أمر "اختر" للتعبير عن أى نوع للربط عن طريق استخدام عمليات علاقية مختلفة داخل أمر "اختر". ويراعى دائمًا فى حالة ظهور اسم العمود فى أكثر من جدول مثل (Dname) أن يؤهل العمود باسم الجدول الملائم لتجنب اللبس. وتتم عملية التأهيل بكتابة اسم الجدول متبوعًا باسم العمود ، على أن يتم الفصل بينهما بنقطة dot.

مثال (٥ – ١٦):

أ- بفرض أن هناك رغبة فى ربط جدول "الإدارة" DEPARTMENT وجدول "المهام" TASK. ويمكن إتمام عملية الربط كالآتى:

```
(i) SELECT  DEPARTMENT.DNAME , LOCATION , MANAGER
    TASKNO ,  TASKNAME.
    FROM  DEPARTMENT , TASK .
    WHERE  DEPARTMENT.DNAME = TASK.DNAME ;
```

(ii)

Dname	Location	Manager	Task No	Task Name
Engineering	12	Al-SALEH	1	Network
Engineering	12	Al-SALEH	2	Math
Training	03	AL-ASKAR	3	Registration
Sales	05	JAFER	4	Big sale
Programs	10	Al-HAJAS	5	Schedules
Library	07	Al-ARFAJ	6	Scripts
Library	07	Al-ARFAJ	7	Books

ب– طباعة أسماء الموظفين فى قسم الهندسة الذين أتموا المهمة :

```
(i) SELECT NAME
    FROM  EMPLOYEE , STATUS
    WHERE  EMPLOYEE.NAME = STATUS.NAME
    AND EMPLOYEE.SSN = STATUS.SSN
    AND EMPLOYEE. DNAME = "ENGINEERING"
    AND ST = "COMPLETED ";
```

(ii)

Name
Salah

تم إظهار القدرة على تجميع الاختيار الرأسى والأفقى فى الأمثلة السابقة باستعمال أمر "اختر" Select. وكذلك تم استعمال أمر "اختر" Select فى عملية الربط. وقد تم تجميع ثلاث عمليات فى أمر واحد كما يتبين من المثال (٥–١٦) . وهذا الاستفسار يبحث عن أسماء الموظفين فى إدارة الهندسة الذين أتموا مهمتهم. ومن ثم يتم ربط جدول "الموظفين" EMPLOYEE وجدول "الحالة" STATUS. الاختيار الأفقى لإدارة الهندسة وإتمام الحالة ثم يليها الاختيار الرأسى للاسم. ويتم الإشارة إلى ربط الجدولين فى عبارة "من" FROM مع أول شرطين فى عبارة "حيث أن" WHERE. وكذلك تتم الإشارة إلى الاختيار الرأسى بواسطة اسم العمود بعد امر"اختر" select. وثم يشار إلى الاختيار الأفقى فى آخر شرطين فى عبارة "حيث إن" WHERE.

الإشارة إلى الفئة الجزئية لعلاقة الربط :

تستخدم عبارة "فى" IN كجزء من عبارة "حيث إن" WHERE للإشارة إلى فئة جزئية لعلاقة الربط. فى حالة الحصول على رقم الضمان الاجتماعى SSN واسم الموظف فى إدارة المبيعات Sales أو إدارة التدريب Training. فإنه يمكن الحصول على الاستفسار مكتوباً بثلاث طرق مختلفة كما هو موضح بمثال (٥–١٧) كالآتى:

(أ) الجزء (a) يعطى نوع الاستعلام الذى سبق شرحه.

(ب) الجزء (b) يوضح استعمال عبارة "فى" IN. فتستعمل عبارة "فى" IN لعضوية فئة، حيث يطلب قيمة {Dname لعضو فى الفئة } "Sales" , "Training" .

(ج) الجزء (c) يستعمل عملية الاتحاد Union للغة الاستعلام البنائية .SQL

مثال (٥-١٧) :

اطبع رقم الضمان الاجتماعى SSN واسم كل موظف Name الذى يوجد إما فى إدارة المبيعات أو إدارة التدريب.

```
(a)  SELECT SSN , NAME
     FROM EMPLOYEE
     WHERE DNAME = "Sales"
     OR DNAME = "Training ";

(b)  SELECT SSN , NAME
     FROM EMPLOYEE
     WHERE DNAME  IN
          ("Sales ","  Training") ;

(c)  SELECT SSN , NAME
     FROM EMPLOYEE
     WHERE DNAME = "Sales"
         UNION
     SELECT SSN , NAME
     FROM EMPLOYEE
     WHERE DNAME =  "Training ";
```

ويكون ناتج عمليات الأجزاء الثلاثة (c) , (b) , (a) كالتالى:

SSN	Name
3333	Mostafa
4444	Sulaiman
8888	Saad

أمر" اختر " المتداخل Nested SELECT :

مثــال (٥-١٨):

بفرض أن هناك رغبة فى الحصول على أسماء المديرين الذين لديهم مهمة تم اتمامها فى أقسامهم. هناك طريقتان لصياغة مثل هذا الاستفسار:

(أ) الطريقة التقليدية للحصول على الربط ، وهى الطريقة السابق شرحها والتى سيتم توضيحها فى الجزء (a) .

(ب) الطريقة الثانية هى صياغة الاستعلام باستخدام أمر "اختر" Select المتداخل كما هو مبين فى الجزء (b) .

```
(a)  SELECT MANAGER
     FROM DEPARTMENT , STATUS
     WHERE DEPARTMENT.DNAME = STATUS.DNAME
     AND ST = "COMPLETED"

(b)  SELECT MANAGER
     FROM DEPARTMENT
     WHERE  DNAME IN
          (SELWCT DNAME
          FROM STATUS
          WHERE ST = "COMPLETED  ") ;
```

ويكون ناتج عمليات الجزأين (b) . (a) هما :

Manager
Al-SALEH
JAFER
AL-ASKER
AL-HAJAS

الدوال التجميعية المثبتة:

تنفذ الدوال التجميعية الخمس المثبتة عملياتها على العمود الذى يتم اختياره من جدول معين يتم تحديده مسبقًا. وهذه الدوال هى:

الدالة	وظيفتها
COUNT	ترجع عدد القيم فى العمود
SUM	ترجع مجموعة القيم العمود
AVG	ترجع متوسط قيم العمود
MAX	ترجع أكبر قيمة فى العمود
MIN	ترجع أصغر قيمة فى العمود

مثال (٢– ١٩):

يبين المثال التالى استخدام الدوال التجميعية المثبتة وكيفية الحصول على النتائج.

(أ) اطبع عدد الإدارات.

(a)

```
(i) SELECT COUNT (DNAME)
    FROM DEPARTMENT ;
(ii) SELECT COUNT (DISTINCT DNAME)
     FROM EMPLOYEE ;
```

ويكون الناتج العائد كالتالى:

(iii)

Result
5

(ب) أطبع أكبر مرتب للموظف فى إدارة التدريب

(a)

(i) SELECT MAX (SALARY)
FROM EMPLOYEE
(ii) WHERE DNAME = "Training ";

ويكون الناتج العائد كالتالى:

(ii)

MAX (Salary
23000

جـ- اطبع مجموع المرتبات فى إدارة المبيعات.

(i) SELECT SUM (SALARY)
FROM EMPLOYEE
WHERE DNAME = "Sales ";

(ii)

SUM (Salary)
53000

د- اطبع متوسط المرتب فى إدارة المبيعات.

(i) SELECT AVG (SALARY)
FROM EMPLOYEE
WHERE DNAME = "Sales ";

(ii)

AVG (Salary)
26500

تجميع النتائج:

كل من عبارتى "جمع حسب" GROUP BY و"يمتلك" HAVING تحققان استفادة بالغة فى اتصالها مع الدوال المثبتة فى لغة الاستعلام البنائية SQL. فعبارة "جمع حسب" GROUP BY تعيد ترتيب الجدول فى مجموعات مبنية على خاصية محددة. ويبين المثال (٥-٢٠) فى الجزء (a) كيفية إيجاد متوسط المرتب فى كل إدارة. ويتم استعمال عبارة "يمتلك" HAVING مع عبارة "جمع حسب" GROUP BY لتصف شرطًا معينًا. ويبين فى الجزء (b) تحديد متوسط المرتب فيما بين ٢٠.٠٠٠ ريال ، ٢٧.٠٠٠ ريال. مع الطباعة فى ترتيب الناتج بناء على اسم الإدارة.

مثال (٥-٢٠):

أ- طباعة متوسط المرتب فى كل إدارة.

(a)

(i)
```
SELSCT DNAME , AVG (SALARY)
FROM EMPLOYEE
GROUP BY DNAME ;
```

(ii)

DNAME	AVG (SALARY)
Eegineering	27500
Sales	26500
Training	23000
Programs	19000
Library	35000

ب- اطبع متوسط المرتب فى كل إدارة لمتوسط المرتب بين ٢٠.٠٠٠ - ٢٧.٠٠٠ريال . على أن يكون الناتج مرتبًا على اسم الإدارة.

(i)
```
SELECT DNAME , AVG (SALARY)
FROM EMPLOYEE
```

```
GROUP BY DNAME
HAVING AVG (SALARY) BETWEEN  2000 AND 27000
ORDER BY DNAME ;
```

Dname	AVG (Salary)
Sales	26500
Training	23000

اختيار الفئة غير الفارغة:

تستخدم عبارة "يوجد" EXISTS لاختيار الفئة غير الفارغة non_empty Set . يبين المثال (٥-٢١) البحث عن لأسماء المهام التى تمت بواسطة الموظف فى الجزء (i) وهو ما يتشابه مع الأمثلة السابقة. فى حين أن الجزء (i) حيث يتم التفكير فيها كالآتى:

يتم أخذ صف من جدول "المهام" TASK ، ثم يليه البحث عن صف فى جدول "الحالة" STATUS التى لها نفس رقم المهمة TaskNo وذات القيمة الحالة ST. ثم يتم اختيار اسم المهمة رأسياً.

وتستخدم علامة النجمة (*) للإشارة إلى جميع الخصائص. وقد يتم استعمال كلمات "أى" ANY و "كل" ALL ، مع عوامل المقارنة التى تسبق الاستعلام الفرعى ، وهو ما سيتم توضيحة فى المثال (٥-٢٢).

مثـال (٥-٢١):

اطبع أسماء كل المهام التى تم إنجازها بواسطة أى موظف:

(i)

```
SELECT  TASKNAME
     FROM TASK , STATUS
     WHERE TASK.TASKNO = STATUS.TASKNO
   AND ST =  "Completed  ";
```

(ii)

```
SELECT TAKNAME
FROM TASK
WHERE EXISTS
(SELECT *
FROM STATUS
 HERE STATUS.TASKNO = TASK.TASKNO
         AND ST = "Completed" ) ;
```

(iii)

Task Name
Math
Registration
Big sale
Schedules

مثـال (٥-٢٢):

أ- اطبع اسم الإدارة واسـم كل الموظفين الذين يتقاضـون مرتبًا أعلى من أى موظف فى إدارة التدريب.

(i)

```
SELECT DNAME , NAME
FROM EMPLOYEE
WHERE SALARY > ANY
(SELECT SALARY
 FROM EMPLOYEE
WHERE DNAME = "Training " ) ;
```

(ii)

DNAME	Name
Eegineering	Mansour
Eegineering	Salah
Sales	Mostafa
Sales	Sulaiman
Library	Ahmed

ب - اطبع اسم الادارة واسم كل الموظفين الذين يحصلون على مرتب أعلى من كل الموظفين فى إدارة "البرامج".

(i)

```
SELECT DNAME , NAME
FROM EMPLOYEE
WHERE SALARY > ALL
                (SELECT SALARY
                  FROM EMPLOYEE
                  WHERE DNAME = 'Programs ');
```

(ii)

Dname	Name
Eegineering	Mansour
Eegineering	Salah
Sales	Mostafa
Sales	Sulaiman
Library	Ahmed
Training	Saad

أنواع تحديث البيانات Types of Data Updates :

هناك ثلاثة أنواع لتحديث البيانات تستعملها لغة الاستعلام البنائية توضح فى المقارنة التالية :

نوع التحديث	الأمر المستخدم
الإضافة	INSERT
الحذف	DELETE
التعديل	UPDATE

(أ) أمر "أضف" INSERT :

* التركيبة اللغوية لأمر "أضف" INSERT

```
INSERT INTO < Table name >
< Column Name >[{ , < Column name> } ... ]
{VALUES ( < literal > ({ , < literal >} ... ] ] }
```

أبسط نوع للإضافة يتضمن إضافة صف فى جدول، كما هو مبين فى المثال (٥-٢٣) الذى يبين كيفية إضافة صف فى جدول "الحالة" STATUS. حيث ليس من الضرورى فى أمر "أضف" INSERT ذكر اسم الجدول فى هذه الحالة. ويتم إضافة قائمة من القيم المناظرة لخصائص العمود بحيث يفصلها عن بعضها فصلة ، مع وضع قائمة هذه القيم بين قوسين.

(ب) أمر "احذف" DELETE :

* التركيبة اللغوية لأمر " احذف" DELETE

```
DELETE FROM < table name >
WHERE < Search Condition >
```

يمكن حذف سطر أو أكثر من الجدول مرة واحدة. على سبيل المثال فى حالة الرغبة فى حذف السطر الذى يحتوى على رقم الضمان الاجتماعى ١١١١ من جدول الموظفين EMPLOYEE وهوما يبينه المثال (٥-٢٣) فى الجزء (b) . ولكن حتى بعد عملية الحذف لا تزال بيانات الموظف فى جدول "الحالة " STATUS ؛ لذا لابد من تكرار عملية الحذف كما هو موضح بالمثال (٥-٢٣) فى الجزء (c).

(ج) أمر "عدل" UPDATE :

النوع الأخير للتحديث هو التعديل ، وهو ما يبينه المثال (٥-٢٣) فى الجزء (b). حيث يوضح كيفية تغيير حالة المهمة "١" فى إدارة الهندسة لرقم الضمان الاجتماعى (١١١١) إلى "midway". ويبين الجزء (e) إضافة (١٠٠٠) ريال لمرتب كل موظف فى إدارة "البرامج" .

*** التركيبة اللغوية لأمر "عدل" UPDATE :**

```
UPDATE < Table name >
SET > Column name > = < value exp. > [{,        } ....]
[ WHERE < Search Condition > ] ;
```

مثال (٥-٢٣):

أ- أضف سطراً (Sales , 4444 , 4 , Completed) فى جدول الحالة .

```
(a) INSERT
    INTO STATUS
   UALUES ('Sales' , ' 4444 ', 4 , ' Completed');
```

ب- احـذف سطراً من جـدول الموظف ، السطر الذى يحـتـوى على رقم الضـمـان الاجتماعى (١١١١).

```
(b) DELETE
     FROM EMPLOYEE
     WHERE SSN = "1111";
```

جـ- احـذف الأسطر التى تحتوى على رقم الضمـان الاجـتمـاعى (١١١١) من جـدول "الحالة" STATUS .

```
(C) DELETE
    FROM STATUS
    WHERE SSN = "1111";
```

د- عدل حـالة رقم المهمة "١" فى إدارة الهندسة لرقم الضـمـان الاجـتـمـاعى (١١١١) إلى "Midway".

```
UPDATE STATUS
SET ST = 'Midway')
WHERE DANAME = 'Engineering'
          AND SSN = "1111"
          AND TASTNO = 1 ;
```

هـ- أضف ١٠٠٠ ريال لمرتب كل موظف فى إدارة البرامج .

```
(a)   UPDATE EMPLOYEE
      SET SALARY = SALARY + 1000
      WHERE DNAME = "Programs ";
```

منظورات لغة الاستعلام البنائية SQL VIEWS :

تسمح لغة الاستعلام البنائية SQL بإنشاء المنظورات التى يتم استنتاجها من الجداول. ويمكن فقط تعريف المنظور على أنه مخزن. وفى كل مرة يتم استخدام المنظور ، يضطر إلى أن ينشأ من جداول فعلية بواسطة نظام قاعدة البيانات؛ لذلك يعكس المنظور التحديث لجداول قاعدة البيانات. وهناك اختلاف بين منظور لغة الاستعلام البنائية والمنظور الخارجى. والمنظور فى لغة الاستعلام البنائية يتم تعريفه بواسطة المستخدم ، ولكن يشبه فكرة المنظورات المعرفة الخرائط التناظرية للجداول التى لا توجد بشكل فردى. الحالة الوحيدة التى يكون فيها منظور لغة الاستعلام البنائية هو منظور خارجى لو كان هناك مستخدم واحد فقط لقاعدة البيانات ، ومدير قاعدة البيانات DBA يقوم بإنشاء كل الجداول التى بعدها يهيأ المنظور المفاهيمى فى البناء المعمارى ذى المستويات الثلاثة Three_Level Architecture .

(أ) إنشاء منظور CREATE VIEWS

فى حالة الرغبة فى إنشاء منظور يحتوى على بيانات الموظفين بدون عنوان الذين مرتبهم لا يقل عن ٣٠,٠٠٠ ريال ، كما هو مبين فى المثال (٥-٢٤) فى الجزء (a). ويتم استخدام أمر "اختر" Select لوصف الاختيار الرأسى وهو من الأوامر المهمة المستخدمة فى إنشاء المنظور.

بعد إنشاء المنظور يمكن صياغة الاستعلامات المطلوبة. كما هو مبين فى المثال (٥-٢٤) فى الجزء (b) الذى يبين الاستفسار الخاص بالبحث عن اسم ومرتب كل موظف فى منظور "الموظفين" VEMP. وتحديث المنظور عادة يكون معقداً وقد يصبح مستحيلاً فى بعض الحالات التى يتم تعريف المنظور فيها بواسطة دالة مثبتة.

كل من ORACLE . DB2 يضع قيوداً على تحديث المنظور، فكلاهما يسمحان فقط بالحصول على المنظور. على سبيل المثال، لو أردنا إضافة صف فى منظور "الموظفين" VEMP لشخص مرتبة يزيد عن ٣٠,٠٠٠ ريال.

هذه الإضافة غير مسموح بها فى حالة استخدام عبارة WITH CHECH OPTION فى تعريف المنظور. خلاف ذلك يسمح بإضافة الشخص فى جدول "الموظفين" EM-PLOYEE ولكنه يختفى عن مستخدم المنظور "الموظفين" VEMP .

* التركيبة اللغوية لأمر إنشاء منظور CREATE VIEW

```
CREATE VIEW < Table name >
< Column name > [ { , Column name > } ..]
AS < query > [ WITH CHECK OPTION ] :
```

(ب) حذف المنظور DROP VIEW :

تحذف المنظورات باستخدام أمر "حذف المنظور" DROP كما هو كما هو مبين في المثال (٥-٢٤) في الجزء (d).

* التركيبة اللغوية لأمر "حذف منظور" DROP VIEW

```
DROP VIEW < Table name >
```

مثال (٥-٢٤):

أ- أنشئ منظوراً للموظفين الذين يتقاضون مرتباً أكبر من ٣٠.٠٠٠ ريال بدون إدراج العنوان.

```
(a)
CREATE VIEW VEMP
AS SELECT DNAME , SSN , NAME . SALARY
FROM EMPLOYEE
WHERE SALARY > 30.000 ;
```

ب- اطبع اسم ومرتب كل موظف في منظور "الموظفين" VEMP

```
(b)
  SELECT NAME , SALARY
  FROM VEMP ;
```

جـ- احذف المنظور "الموظفين" VEMP

```
(C)
  DROP VIEW VEMP ;
```

ثالثاً : التحكم في البيانات Data Control Language (DCL) :

(أ) منح الصلاحيات GRANT :

يكمن منح الصلاحيات في كل من المنظورات والجداول في السماح باسترجاع - فقط - بيانات جدول أو منظور Select ، وتحديث بيانات الجداول والمنظورات Updates وتغيير بناء جدول Alter أو استخدام كل مستويات التداول (٥) ALL.

أمر "منح الصلاحية" GRANT :

* التركيبة اللغوية لأمر "منح الصلاحية" GRANT :

```
GRANT < STATEMENT >
ON { < table name > | < view name > }
TO { < user ID > [{ , < user ID >} ... ] |
      PUBLIC }
      [WITH GRANT OPTION] ;
```

(ب) سحب الصلاحية REVOKE :

* التركيبة اللغوية لأمر "سحب الصلاحية" REVOKE :

```
REVOKE < statement >
ON     { < Table name > | < View name > }
FROM { < User ID > [ { , > User ID >} ... ] |
       PUBLIC } ;
```

مثال (٥-٢٥):

أ- امنح صلاحيات الاسترجاع والتعديل لعمود المرتب لمدير الشئون المالية الذى يحمل الرقم المعرف 418396 .

(a)

```
GRANT SELECT , UPDATE (Salary)
ON      EMPLOYEE
TO       418396 ;
```

ب- امنح صلاحيات الاسترجاع والتحديث لمنظور جدول الموظفين، عدا عمود المرتبات لموظفى الإدارة المالية التى يحمل موظفوها أرقام المعرفات التالية :

418290, 418190, 418090

(b)

```
GRANT SELECT , VPDATE , DELETE , INSERT
ON      VEMP
TO       418290 , 418190 , 418090 ;
```

جـ- امنح صلاحيات استرجاع بيانات منظور "الموظفين" VEMP لكل مستخدمى قاعدة البيانات.

(C)

```
GRANT SELECT
ON  VEMP
TO  PUBLIC ,
```

د- اسحب صلاحيات الاسترجاع والتحديث لمنظور "الموظفين" من موظفى الإدارة المالية.

(d)

```
REVOKE   ALL
ON         VEMP
FROM      418290, 418190, 418090;
```

هـ- اسحب صلاحيات استرجاع بيانات منظور "الموظفين" VEMP من مستخدمى قاعدة البيانات

```
REVOKE SELECT
ON  VEMP
FROM  PUBLIC ;
```

الهوامش :

1. [KORTH, 1986], Henry F. Korth and Abraham Silberschatz, **Database System Concepts**, International Edition, McGraw-Hill, Inc., 1986.
2. [ULLMAN,1988], Ullman J.D., **Principles of Database and Knowledge-Base System**, Vol. 1, Computer Science Press, 1988.
3. [DATE,1990], C.J. Date, **An Introduction to Database Systems**, Volume I, Fifth Edition, Addison-Wesley Publishing Company Inc.,1990.
4. [ELMASRI, 1989], Ramez Elmasri and Shamkant B. Navathe, **Fundamentals of Database Systems**, Benjamin/Cummings, Redwood City, Calif., 1989.
5. [Connolly 1996], Thomas. M. Connolly, Carolyn E. Begg, **Database Systems**, Addison-Wesley Publishing Co., Inc., 1996.
6. [Mcallister 1998], Mcallister, Andrew J. and Shorpe David, 'An Approach for Decomposing N-ary Data Relationships ', **Software-Practice and Experience**, Vol. 28(2), Feb.,1998.
7. [DATE,1995], C.J. Date, **An Introduction to Database Systems**, Volume I, Sixth edition, Addison-Wesley Publishing Company Inc.,1995.
8. [GRANT, 1987], John Grant, **Logical Introduction to Databases**, Harcourt Brace Jovanovich, Publishers and its subsidiary, Academic Press, 1987.
9. [Brathwaite 1991], Dr. Kenmore S. Brathwaite, **Relational Databases, Concepts, Design, and Administration**, McGraw-Hill Inc. New York,1991.

الفصل السادس

نماذج البيانات الدلالية (اللفظية)



مقدمة :

تعد نماذج البيانات الدلالية من أهم النماذج التى يرتكز عليها مصممو قواعد البيانات. وقد تضمنت نماذج البيانات الدلالية معظم مفاهيم الاتجاه الشيئى؛ مما ساعد فى ظهور نماذج البيانات الشيئية الموجهة. وسوف يتم فى هذا الفصل استعراض الموضوعات التالية:

نماذج البيانات الدلالية فى النماذج التقليدية:

منذ البداية تغلغلت نماذج البيانات الدلالية فى نماذج البيانات الأولية وكذلك التقليدية سواء التبحرية منها أو العلاقية. وقد سبق استعراض تفاصيل تصنيفات نماذج البيانات التقليدية سواء التبحرية منها أو العلاقية فى الفصول السابقة. وسوف يتم استعراض تفاعل نماذج البيانات الدلالية فى هذه النماذج.

نموذج البيانات المفاهيمى العالى المستوى:

لقد سبق التطرق فى الفصل الأول لتعريف التصميم المفاهيمى الذى ينتج عن متطلبات المستفيدين ونشاط تحليل مجموعة متطلبات قواعد البيانات التى ينبغى احتواؤها بواسطة نموذج البيانات. وإنه من الأهمية بمكان استخدام نموذج البيانات المفاهيمى أو الدلالى ذى المستوى العالى؛ لخدمة مرحلة تحليل البيانات فى بدء طريقها التمهيدى.

المفاهيم الأساسية للنماذج المنطقية:

هناك العديد من المفاهيم التى ينبغى طرحها قبل البدء فى شرح مكونات نموذج كينونة – علاقة ER. من هذه المفاهيم مفهوم الكينونة، ونوع الكينونة وأنوا ع علاقات الربط وتدويناتها وتعدديتها. حيث تشكل هذه المفاهيم البنية الأساسية للنماذج الدلالية.

نموذج كينونة – علاقة ER :

يعتبر نموذج كينونة – علاقة من أكثر النماذج إفادة لمصممى قواعد البيانات؛ لأنه يؤكد على تمثيل المخططات وليس الوقائع التى تتغير بشكل متكرر. وإن تهيئة النموذج

تعطى نظم إدارة قواعد البيانات القدرة على التنفيذ المباشر لقاعدة البيانات التى تم تصميمها من خلال المخطط المفاهيمى العالى المستوى.

تفكيك علاقات الربط:

تشكل علاقات الربط فى نموذج كينونة – علاقة إحدى المشاكل كثيرة التكرار، ولا سيما عند نمذجة علاقة ربط ذات الدرجة الثالثة أو أكثر. والسؤال الحتمى: كيف يمكن تحديد علاقات ربط ثنائية أبسط وكافية؟ وللإجابة عن هذا السؤال: يمكن اتباع طريقة "الخطوات الثمان" المستخدمة فى تفكيك علاقات الربط ذات الدرجة الثالثة فأكثر.

طريقة الخطوات الثمان:

هذه الطريقة تم تصميمها لاتخاذ قرار تفكيك علاقة الربط R الفردية ، ذات الدرجة N. ويتم تحليل علاقة الربط خلال هذه الطريقة عبر ثمانى خطوات. يتم تتبع كل من هذه الخطوات بشكل منفصل عن الآخر. وسوف يتم توضيح ذلك من خلال العديد من التطبيقات.

نموذج كينونة – علاقة المطور EER :

يعد هذا النموذج تطويراً لنموذج كينونة – علاقة ، حيث يتم تمثيل معظم المفاهيم المهمة فى نمذجة البيانات الدلالية بشكل مناسب. ومن الإضافات المهمة لذلك النموذج: الأنواع الفرعية، وراثة الخاصية، بالإضافة إلى توظيفه لمفهوم التجريد.

المبادئ الأساسية لنمذجة البيانات الدلالية لتطبيقات قواعد البيانات:

سوف يتم استعراض المفاهيم التجريدية التى تستخدم فى نماذج البيانات الدلالية. بعضٌ من هذه المفاهيم مزدوج، وكل عكس الآخر مثل: التصنيف/التفريع، التعميم/التخصيص. والبعض الآخر يشكل المفاهيم المترابطة مثل: التجميع والارتباط بالإضافة إلى مفهوم التعريف.

نماذج البيانات الوظيفية:

تستخدم نماذج البيانات الوظيفية مفهوم "الوظيفة الرياضية" كبناء نمذجة أساسى. وتعرض هذه النماذج الكينونات والوظائف عليها ككتل بنائية أساسية. وتعد هذه النماذج أحد نماذج البيانات الدلالية.

نماذج البيانات الدلالية فى النماذج التقليدية:

يعتبر تصميم قواعد البيانات أساسًا مهمًا فى نمذجة البيانات. نموذج البيانات هو البناء الهيكلى للبيانات؛ لذا كان لابد من التمييز بين أنوا ع نماذج البيانات التقليدية.

١- نماذج البيانات الأولية:

فى هذه النماذج يمكن تمثيل الأشياء objects بهياكل سجلات مجمعة فى هيكل ملف. وكانت العمليات الرئيسية المتوافرة هى عمليات قراءة وكتابة السجل.

٢- نماذج البيانات التقليدية :

(أ) النماذج التبحرية:

يعتبر نموذج البيانات الهرمى امتداداً لنموذج البيانات الأولى (البدائى). ويعتبر نموذج البيانات الشبكى امتداداً لنموذج البيانات الهرمى ، وكلاهما أيضاً اعتمد على طريقة تمثيل الأشياء objects بهياكل سجلات مجمعة فى هيكل ملف.

(ب) النماذج العلاقية:

انطلق نموذج البيانات العلاقى أساسًا من الطرق المتنوعة لفهم نماذج البيانات الهرمية والشبكية ، واعتمد على السجل ككينونة فى بناء الجدول العلاقى الذى يمثل ملفًا. والسبب الرئيسى وراء تفضيل إنشاء تطبيقات مبنية على النموذج العلاقى عن غيره من النماذج الأخرى هو بساطة مفهوم وراثة النموذج، حيث إن أساس النموذج هو الجدول العلاقى. ومع ذلك فإنه من المستحيل مع النموذج العلاقى الأساسى الحصول والتحكم فى الدلاليات الكثيرة لتطبيق معقد بهذا الإطار البسيط. على سبيل المثال: خلال النموذج العلاقى يتم فرض السلامة المرجعية بشكل إجبارى؛ لأنه لا توجد آلية للتمييز بين مختلف أنوا ع علاقات الربط التى قد توجد بين الكينونات ، مثل علاقات الربط (متعدد - لمتعدد) وعلاقات الربط (واحد - لمتعدد) أو وضع تصور للصفات الموجودة والمعتمدة على علاقات الربط. وعندما يوجد مثل هذا التمييز، فإنه من الممكن تعريف دلاليات لعمليات إنشاء وحذف علاقات الربط لمختلف الحالات. وقد تم تعريف نماذج جديدة امتداداً للنموذج العلاقى أكثر تعبيراً وتؤدى إلى توفير دلاليات أكثر

لنطاق التطبيق داخل النموذج. وأحد هذه النماذج التى تم توافرها لتسمى نماذج البيانات الدلالية قد تم تعريفة بواسطة كود Codd وعرف باسم RM/T. وهذا النموذج يحتوى على بعض الكينونات التى تمثل الأشياء للعالم الحقيقى. والأشياء تتضمن علاقات الربط بين الأشياء الأخرى. وتتضمن الأشياء علاقات الربط وتنمذج ككينونات.

٣- نماذج البيانات الدلالية:

كانت المشكلة الرئيسية فى النماذج التقليدية هى احتفاظها باتجاه السجل بشكل أساسى. وهذا يعنى أن دلاليات المعلومات (أى معانيها) داخل قواعد البيانات قد انسدلت وأصبحت غير ميسرة للبروز من داخل قواعد البيانات نفسها. ويجب أن تكون المعلومات الدلالية مدركة من قبل المستفيد الذى يستخدم الطرق المتنوعة لفهم قواعد البيانات. ولهذا السبب ، فإن عدداً من نماذج البيانات الدلالية قد تم اقتراحه لمحاولة توفير معانى أكثر تعبيرية لتمثيل معنى المعلومات بشكل أكثر مما هو متوافر فى النماذج التقليدية.

نموذج البيانات المفاهيمى العالى المستوى:

ينتج عن متطلبات المستفيدين ونشاط تحليل مجموعة متطلبات لقواعد البيانات التى يجب أن يتم الاستيلاء عليها بواسطة نموذج البيانات فى المرحلة التالية ، وهى ما تعرف بالتصميم المفاهيمى الذى سبق التطرق إليه وتوضيحه كما فى الشكل رقم (١-١) بالفصل الأول. ويفضل استخدام نموذج بيانات مفاهيمى أو نموذج البيانات الدلالى ذى المستوى العالى لخدمة مرحلة التحليل. وإنه من الأهمية بمكان استعمال ذلك النموذج فى تلك المرحلة: لأن قواعد البيانات تكون فى بدء طريقها التمهيدى بواسطة مجموعة كاملة من المستخدمين غير المرتبطين بنموذج تنفيذى محدد ولا بنظام إدارة قواعد بيانات معين.

من العيوب الشائعة فى المنظمات بخصوص نشاط تصميم قواعد البيانات الافتقار إلى الاهتمام بتصميم قواعد بيانات مفاهيمية، فى حين يتركز الاهتمام على بعض نظم إدارة قواعد البيانات المستهدفة. ويزيد مصممى قواعد البيانات من أهمية نشاط تصميم قواعد البيانات المفاهيمية بإقرار بعض المعرفة غير الملائمة للنموذج العلاقى

والتى تكون أكثر تعبيراً. ومن ثم تؤدى إلى الحصول على دلاليات أكثر فى مجال التطبيق داخل النموذج. ومثل هذه النماذج يطلق عليها نماذج البيانات الدلالية.

وقد أسهمت نماذج البيانات الدلالية فى اتجاهين لتطوير نماذج البيانات الشيئية الموجهة. أحدهما: لوصف البناء المعمارى لنظم قواعد بيانات متقدمة. والآخر: لتحليل وتصميم نظم قواعد البيانات.

ففى الاتجاه الأول: أنتجت نماذج البيانات الدلالية كثيراً من التطويرات فى المجموعة الحالية لنظم إدارة قواعد البيانات الشيئية الموجهة ، مثل أونتوس ONTOS وإيريس IRIS وجمستون أوبال Gemstone OPAL. أما فى الاتجاه الثانى : أنتجت نماذج البيانات الدلالية فرعاً لمنهجية التحليل الشيئى الموجه. وقد طور كل من كواد Coad ويوردون Yourdon طريقة لفهم الواقعة instance ، والتى تأثرت كثيراً بطريقة نموذج كينونة - علاقة ER. ومع ذلك ، فإن معظم المفاهيم المهمة فى نمذجة البيانات الدلالية يمكن أن تمثل بشكل ملائم فى نموذج كينونة-علاقة المطور EER. ويتضمن الأخير إضافات مهمة للنموذج الأصلى والتى تشتمل على مفهوم النوع الفرعى Sub-class والنوع الأصلى Superclass ، وكذلك مفهوم توارث الخاصية. وأحد الأسباب الرئيسية التى أدت إلى الانتشار الواسع له هو قدرته على توفير تقنية أعلى-أسفل Top-Down لتصميم قاعدة البيانات وكذلك توظيفه لمفهوم التجريد Abstraction.

يعرف نموذج البيانات بأنه مجموعة المفاهيم التى تساعد فى توصيف قاعدة البيانات ومجموعة العمليات المترابطة لتوصيف الاسترجاع والتحديث على قاعدة البيانات. ويبين الشكل رقم (١-١) توصيف مبسط لعملية تصميم قاعدة البيانات طبقاً للخطوات التالية [١]:

١- مجموعة المتطلبات والتحليل:

فى هذه الخطوة يعمل مصممو قواعد البيانات على مقابلات المستفيدين لفهم وتوثيق متطلبات بياناتهم. وينتج عن هذه الخطوة توثيق متطلبات المستفيدين بأسلوب موجز. مع مراعاة توصيف هذه المتطلبات فى شكل كامل وتفصيلى على قدر الإمكان. وبمجرد إتمام تجميع وتحليل المتطلبات يمكن العمل خلال المخطط المفاهيمى.

٢- المخطط المفاهيمي:

ويمثل إنشاء المخطط المفاهيمي الخطوة التالية لتجميع وتحليل المتطلبات لقاعدة البيانات باستعمال نموذج البيانات المفاهيمي العالي المستوى. وتسمى هذه الخطوة بتصميم قاعدة البيانات المفاهيمية. والمخطط المفاهيمي هو توصيف موجز لمتطلبات بيانات المستفيدين بحيث تحتوى على تفاصيل توصيفات أنواع البيانات وعلاقات الربط والقيود. ويتم التعبير عن هذه التوصيفات باستخدام المفاهيم المتوافرة في نموذج البيانات العالي المستوى؛ لأن هذه المفاهيم غير مضمنة في تفاصيل التنفيذ. وأن مصممي قاعدة البيانات يركزون على توصيف خصائص البيانات دون الخوض في تفاصيل التخزين.

٣- الخريطة التناظرية لنموذج البيانات:

وتمثل هذه الخطوة التنفيذ الفعلي لقواعد البيانات باستعمال نظم إدارة قواعد بيانات تجارية. ومعظم هذه النظم تستعمل نموذج البيانات التنفيذي ، وهو يمثل تحويل المخطط المفاهيمي من نموذج البيانات العالي المستوى إلى نموذج البيانات التنفيذي.

٤- التصميم المادي لقاعدة البيانات:

ويتم في هذه الخطوة توصيف هياكل التخزين الداخلية وتنظيم الملفات الخاصة بقواعد البيانات.

مثال (٦-١):

يصف هذا المثال قاعدة بيانات شركة المشاريع الهندسية (الافتراضية) Engineering Projects Company لتوضيح عملية تصميم قاعدة البيانات. وسوف يتم سرد متطلبات البيانات لتلك القاعدة وإنشاء مخططها المفاهيمي تدريجياً ، مع تقديم مفاهيم النمذجة لنموذج كينونة - علاقة. وتتضمن هذه القاعدة ملفات لموظفي الشركة Employees والإدارات Department والمشاريع Projects. وبعد مرحلة تجميع المتطلبات وتحليلها ، سيتم سرد توصيف العالم الخارجي للشركة من قبل مصممي قاعدة البيانات؛ لكي يتم تمثيله في قاعدة البيانات في قائمة المتطلبات التالية:

١- يتم تنظيم الشركة في إدارات Departments. كل إدارة لها اسم DName ورقم DNum ومدير Dmanager، ويتم تحديد تاريخ بدء عمل مدير الإدارة MDate. وقد يكون للإدارة عدة مواقع .Dlocations.

٢- تراقب الإدارة عدداً من المشاريع Project. وكل مشروع له اسم PName ورقم PNum ومكان محدد PLocation.

٣- يتم تخزين اسم الموظف Ename ورقم الضمان الاجتماعي SSN والعنوان Eaddr والمرتب Salary والجنسية Nationality وتاريخ الميلاد BDate. وكل موظف يعمل بإدارة واحدة فقط ولكن قد يشارك في العديد من المشاريع والتي ليس بالضرورة أن يتم مراقبتها من قبل الإدارة التي يعمل بها. ويتم الاحتفاظ بعدد ساعات عمل الموظف في الأسبوع بالنسبة لكل مشروع. ويتم الاحتفاظ أيضاً بمشرف لكل موظف.

٤- يتم الاحتفاظ بأسماء من يعولهم الموظف لأغراض التأمين، وجنسهم، وتاريخ ميلادهم، وعلاقتهم بالموظف.

المفاهيم الأساسية للنماذج المنطقية:

(أ) الكينونة:

عبارة عن شخص أو مكان أو شيء أو حدث أو مفهوم في بيئة المستفيدين، تحتاج المؤسسة أن تجمع بيانات عنه وتخزنها. ويتم تمثيل الكينونة في النماذج التبحرية بالسجلات. وهذا يعني أن سجل الموظف هو كينونة الموظف في حين يتم تمثيل الكينونة في النموذج العلاقي بالقيم المرتبة tuple. حيث إن القيم المرتبة على سبيل المثال لموظف معين في النموذج العلاقي هي كينونة ذلك الموظف[٢].

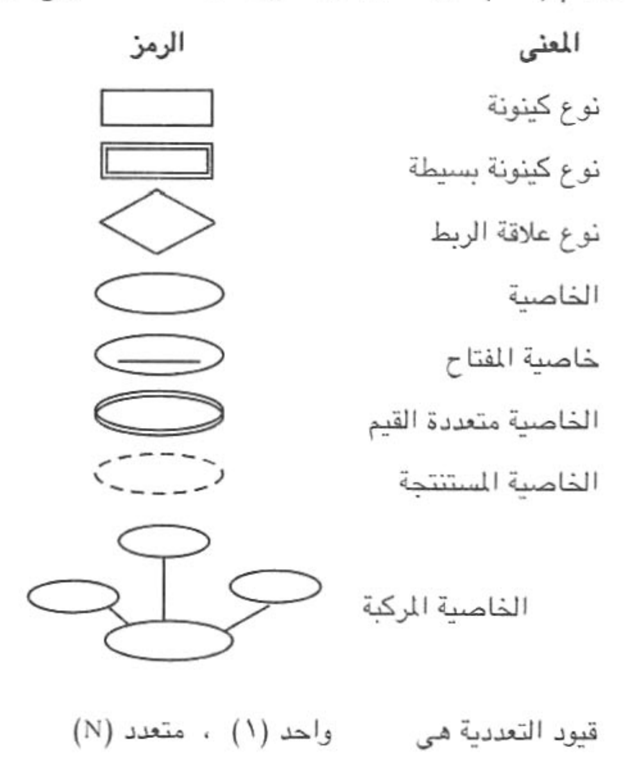
(ب) نوع الكينونة:

هي مجموعة كينونات لها خواص مشتركة. ويتم تمثيل نوع الكينونة في النماذج التبحرية بنوع السجل (الملف) في حين يتم تمثيل نوع الكينونة في النموذج العلاقي بالجدول العلاقي (العلاقة relation). أي أن الجدول العلاقي الخاص "بالموظفين" هو نوع كينونة "الموظفين"[٢].

(ج) تدوينات أنواع علاقات الربـط Relationships:

الأشكال المعتادة لنموذج كينونة – علاقة تستعمل نوع تشين Chen Style لتوصيف علاقات الربط . وفيما يلى تدوينات ورموز تشين Chen Style المستخدمة[١]:

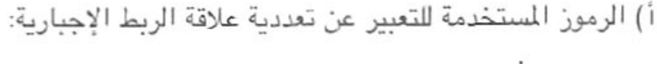
شكل رقم (٦–١) تدوينات ورموز تشين Chen Style لنموذج كينونة – علاقة



قيود التعددية هى واحد (١) ، متعدد (N)

وتصف قيود التعددية أنواعاً أخرى عديدة لنمذجة نموذج – علاقة ER. والمفتاح هو كيفية فهم تحديد قيود التعددية فى التدوين المستخدم باستخدام تدوينات ميرز Merise أو تدوينات هندسة المعلومات Information Engineering أو تدوينات علاقات الربط الثنائية. وفيما يلى تدوينات ورموز هندسة المعلومات [٤،١٢] Information Engineering المستخدمة لتوصيف التعددية بنموذج كينونة – علاقة ER ويستخدم فيها رمز "قدم الغراب" للتعبير عن "متعدد":

أ) الرموز المستخدمة للتعبير عن تعددية علاقة الربط الإجبارية:



علاقة ربط واحد إجبارية

علاقة ربط متعدد إجبارية

علاقة ربط (واحد/متعدد) إجبارية

ب) الرموز المستخدمة للتعبير عن تعددية علاقة الربط الاختيارية:

علاقة ربط (صفر/واحد) اختيارية

علاقة ربط (صفر/متعدد) اختيارية

أنواع علاقات الربط Relationships :

علاقة الربط هى ارتباط بين كينونات نوع كينونة أو أكثر ، أى الارتباط بين سجلات نوع سجل معين أو مجموعات القيم المرتبة لجدول علاقى معين أو أكثر[٢]،[٣].

(١) علاقة الربط واحد – لواحد One-to-one:

وهى تعنى ارتباط كينونة معينة فى نوع كينونة ما بكينونة مناظرة فى نوع كينونة أخرى. مثال ذلك ، نوعا كينونة "الموظف" و "الإدارة". يتم ارتباط كينونة الموظف (فقط المدير) فى نوع كينونة "الموظف" بكينونة إدارة فى نوع كينونة "الإدارة" فى حالة وجود مدير واحد يشرف على الإدارة. ويمثل هذه العلاقة بالرسم التخطيطى فى الشكل رقم (٦-٢).

شكل رقم (٦ – ٢) النموذج المنطقى لعلاقة الربط واحد – لواحد بين نوعى كينونة "الموظف" "والادارة"

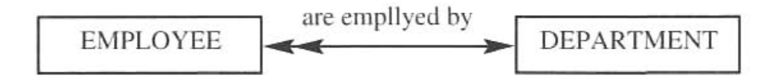


(٢) علاقة الربط واحد – لمتعدد One-to-many :

وهى تعنى ارتباط كينونة معينة فى نوع كينونة ما بكينونة أو أكثر فى نوع كينونة أخرى. مثال ذلك: فى نوعى كينونة "الموظف" و "الإدارة" ، قد يتم ارتباط كينونات أكثر

من موظف بكينونة إدارة واحدة. وتمثل هذه العلاقة بالرسم التخطيطى فى الشكل رقم (٦-٣).

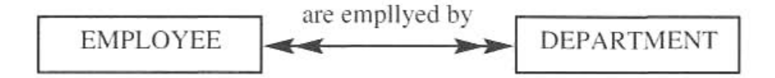
شكل رقم (٦-٣) النموذج المنطقى لعلاقة الربط واحد-لمتعدد بين نوعى كينونة "الموظف" و"الادارة"



(٣) علاقة الربط متعدد – لمتعدد many-to-many :

وهى تعنى ارتباط كينونة أو أكثر من كينونة فى نوع كينونة ما بكينونة أو أكثر فى نوع كينونة أخرى. مثال ذلك: فى نوعى كينونة "الموظف" و "الإدارة" ، قد ترتبط كينونات موظف أو أكثر بكينونة إدارة أو أكثر فى حالة ما إذا كان الموظف يعمل بأكثر من إدارة، والإدارة الواحدة بها أكثر من موظف. وتمثل هذه العلاقة بالرسم التخطيطى فى الشكل رقم (٦-٤).

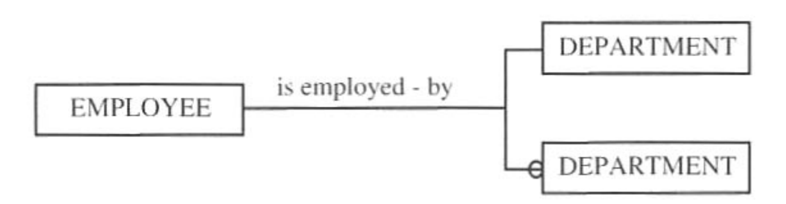
الشكل رقم (٦-٤) النموذج المنطقى لعلاقة الربط متعدد – لمتعدد بين نوعى كينونة "موظف" و"الإدارة"



(٤) علاقة الربط المقصورة Mutually exclusive :

وهى تعنى ارتباط كينونة من نوع كينونة ما بأى من كينونات نوع كينونة أخرى وليس بأكثر من كينونة فى نفس الوقت. مثال ذلك: قد الموظف يعمل إما بالإدارة (A) أو بالإدارة (B) ولكن ليس بكليهما. وعلاقة الربط فى مثل هذه الحالة بين كينونة الموظف وأى من كينونات الإدارة تسمى بالمقصورة. وتمثل هذه العلاقة بالرسم التخطيطى فى الشكل رقم (٦-٥) حيث يشير رمز الدائرة الصغيرة (o) على الخط الواصل بين أنواع الكينونات إلى أن العلاقة اختيارية Optional. وخلاف ذلك تكون العلاقة إجبارية mandatory.

الشكل رقم (٦- ٥) النموذج المنطقى لعلاقة الربط المقصورة بين نوعى كينونة "الموظف" و"الإدارة" لكينونتين (A) أو (B).



(٥) علاقة الربط الشاملة Mutually inclusive:

وهى تعنى وجوب ارتباط كينونة من نوع كينونة ما بأكثر من كينونة من نوع كينونة أخرى فى نفس الوقت. مثال ذلك : كينونة الموظف قد ترتبط بكل من كينونتى الإدارة (B) ، (A) فى نفس الوقت. وتسمى علاقة الربط فى هذه الحالة بعلاقة الربط الشاملة [٤]،[٦]. وتمثل علاقة الربط بالرسم التخطيطى كما فى الشكل رقم (٦-٦).

الشكل رقم (٤-٦) النموذج المنطقى لعلاقة الربط الشاملة بين نوعى كينونة "الموظف" و"الإدارة" لكينونتين (A) ، (B)



(٦) علاقة الربط الإجبارية Mandatory:

وهى تعنى وجوب ارتباط كينونة معينة من نوع كينونة ما بكينونة من نوع كينونة أخرى. مثال ذلك: فى بعض الأحيان قد يقرر صاحب العمل Employer وجوب وجود الإدارة قبل تخصيص الموظف لها. وعلاقة الربط فى هذه الحالة هى علاقة إجبارية. وتمثل علاقة الربط الإجبارية بالرسم التخطيطى فى الشكل رقم (٦-٧).

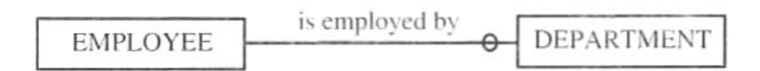
الشكل رقم (٦-٧) النموذج المنطقى لعلاقة الربط الإجبارية بين نوعى كينونة "الموظف" و"الإدارة"

EMPLOYEE — is employed by — DEPARTMENT

(٧) علاقة الربط الاختيارية Optional :

تعنى عدم وجوب ارتباط كينونة معينة من نوع كينونة ما بكينونة من نوع كينونة أخرى. مثال: قد يخصص الموظف لإدارة معينة قبل وجودها وتسمى علاقة الربط فى مثل هذه الحالة بعلاقة الربط الاختيارية وتمثل علاقة الربط بالرسم التخطيطى فى الشكل رقم (٦-٨).

الشكل رقم (٦-٨) النموذج المنطقى لعلاقة الربط الاختيارية بين نوعى كينونة "الموظف" و"الإدارة"



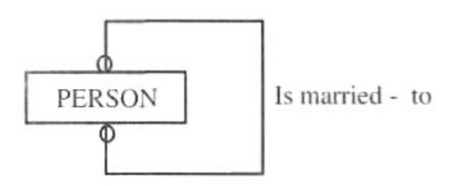
(د) درجة علاقة الربط Arity of Relationship :

وهى تمثل عدد أنواع الكينونات التى تشارك فى علاقة الربط. ومعظم درجات علاقات الربط الشائعة هى، الأحادية Unary والثنائية Binary والثلاثية Ternary.

(١) درجة علاقة الربط الأحادية:

وهى تمثل الارتباط بين الكينونات الموجودة فى نفس الكينونة. وغالبًا ما تعرف بعلاقة الربط التى تعيد تعريف نفسها. على سبيل المثال: علاقة الربط "متزوج من" Married-To. وهى غالبًا ما تكون (واحد - لواحد) فى نوع كينونة "شخص" PERSON على نفس الكينونات (مجموعات القيم المرتبة) المكونة لها كما هو موضح بالشكل رقم (٦-٩أ). وأيضًا علاقة الربط "يدير" Manages فى نوع كينونة "الموظف" EMPLOYEE قد تكون (واحد - لمتعدد). وقد تكون اختيارية أو إجبارية كما هو موضح بالشكل رقم (٦-٩ب).

الشكل رقم (٦-٩أ) يوضح درجة علاقة الربط الأحادية (اختيارية)



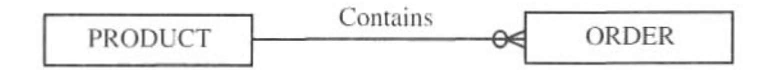
الشكل رقم (٦-٩ب) يوضح درجة علاقة الربط الأحادية (اختيارية - إجبارية)



(٢) درجة علاقة الربط الثنائية:

وهى تمثل الارتباط بين الكينونات الموجودة فى نوعى كينونة مختلفتين. على سبيل المثال : علاقة الربط "يحتوى على" Contains بين نوعى كينونة "المنتج" PRODUET و"الطلبات" ORDER. حيث يتم إنتاج منتج معين بناء على أكثر من طلب. وقد تكون اختيارية كما هو موضح بالشكل رقم (٦-١٠).

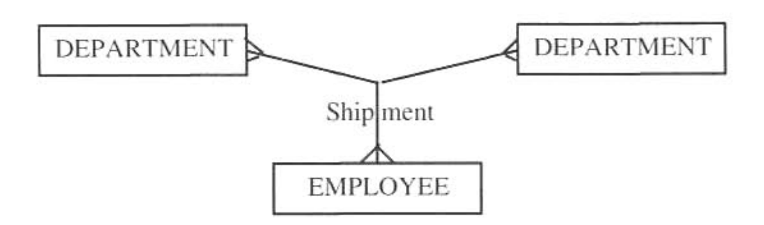
الشكل رقم (٦-١٠) يوضح درجة علاقة الربط الثنائية



(٣) درجة علاقة الربط الثلاثية:

وهى تمثل علاقة ربط متزامنة (آنية) بين ثلاث أنواع كينونات مختلفة. على سبيل المثال: علاقة الربط "شحن" Shipment بين نوع كينونة "منتج" PRODUCT ونوع كينونة "البائع" VENDOR ونوع كينونة "المستودع" WAREHOUSE كما هو موضح بالشكل رقم (٦-١١).

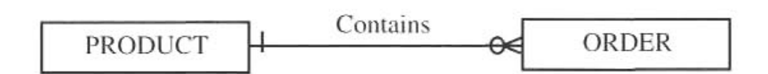
الشكل رقم (٦-١١) يوضح درجة علاقة الربط الثلاثية



(هـ) التعددية فى علاقات الربط:

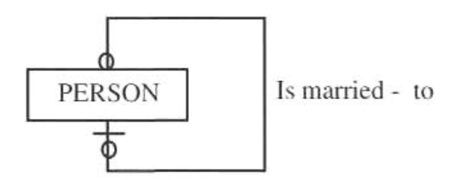
هى عدد الكينونات (مجموعات القيم المرتبة) من نوع كينونة معينة التى يمكن (أو يجب) أن ترتبط مع كل كينونة (قيمة مرتبة) من نوع الكينونة المرتبطة معها . على سبيل المثال : التعددية فى علاقة الربط "تقديم الطلبات" Places بين نوعى كينونة "العميل" CUSTOMER "والطلبات" ORDER تبين أن للعميل طلبات متعددة (يكتب الرقم أسفل علاقة التعدد فى حالة معرفة عدد الكينونات بالضبط) وهى إجبارية ـ اختيارية [٦] كما هو مبين فى الشكل رقم (٦-١٢).

الشكل رقم (٦-١٢) تعددية إجبارية (العميل) ، اختيارية (الطلبات)



وفى المثال الخاص بنوع كينونة "الشخص" PERSON بخصوص علاقة الربط "يتزوج من" Married-To فإن التعددية اختيارية فى علاقة ربط أحادية ، كما هو موضح بالشكل رقم (٦-١٣).

الشكل رقم (٦-١٣) تعددية اختيارية



وأحياناً تحدد عدد الكينونات بشكل دقيق على التمثيل المنطقى لعلاقة الربط. على سبيل المثال : فى علاقة الربط "يخصص" بين نوعى كينونة "موظف" EMPLOYEE و"المشروعات" PROJECT ، يمكن إيضاح عدد الكينونات بدقة لو فرض أن الموظف الواحد يخصص له خمسة مشروعات اختيارية ، كما هو مبين فى الشكل رقم (٦-١٤).

الشكل رقم (٦-١٤) يوضح التعددية بدقة.



نموذج كينونة - علاقة ER :

وهو نموذج يمثل كلاً من أنوا ع الكينونات وأنوا ع علاقات الربط ، ويؤكد على تمثيل المخططات وليس على الوقائع. ويعد نموذج كينونة - علاقة أكثر إفادة لمصممى قاعدة البيانات حيث إن مخطط قاعدة البيانات نادراً ما يتغير فى حين أن الوقائع قد تتغير بشكل متكرر . وعادة من السهل أن يتم عرض المخطط عن عرض وقائع قاعدة البيانات لأنه بسيط جداً . ونموذج كينونة - علاقة هو نموذج بيانات مفاهيمى عالى المستوى، حيث إن نموذج البيانات يمكن أن يعرف كمجموعة من مفاهيم التى تساعدنا على توصيف هيكل قاعدة البيانات ومجموعة العمليات المرتبطة بها لتوصيف الاسترجا ع والتحديث لقاعدة البيانات.

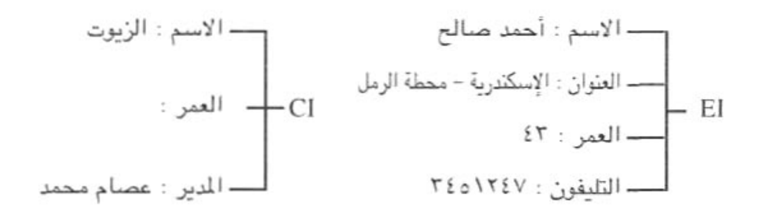
وفى الوقت الحاضر يتم استعمال نموذج كينونة - علاقة بشكل أساسى أثناء عملية تصميم قاعدة البيانات، حيث إن تهيئة النموذج تعطى نظم إدارة قواعد البيانات القدرة على التنفيذ مباشرة لقاعدة البيانات التى تم تصميمها من خلال ذلك المخطط المفاهيمى عالى المستوى. تمثل معلومات نموذج كينونة - علاقة مجموعة من المفاهيم الدلالية [١] التالية:

١- الكينونات والخصائص Entities & Attributes :

الهدف الأساسى لنموذج كينونة - علاقة هو تمثيل **الكينونة** . ولكل كينونة خواص معينة تسمى **الخصائص** والتى تصف الكينونة. وعلى سبيل المثال كينونة "موظف" قد

يتم توصيفها بواسطة مجموعة خصائص هى: اسم الموظف ، العمر ، العنوان ، المرتب والوظيفة. وكل كينونة تحتوى على قيمة لكل خاصية. وقيم الخاصية التى تصف كل كينونة تصبح جزء رئيسياً من البيانات المخزنة فى قاعدة البيانات. ويبين الشكل رقم (٦-١٥) كينونتى "الموظفE1" و"الشركة C1" وخصائصهما، بحيث أن كينونة الموظف E1 لها أربعة خصائص هى: الاسم ، العنوان ، العمر ، التليفون. وقيم هذه الخصائص هى: أحمد صالح ، الإسكندرية محطة الرمل ، ٤٣ ، ٣٤٥١٢٤٧" على التوالى . وكينونة الشركة C1 لها ثلاثة خصائص هى: "الاسم ، المركز الرئيسى ، المدير". وقيم هذه الخصائص هى: "الزيوت ، الإبراهيمية ، عصام محمد"، على التوالى.

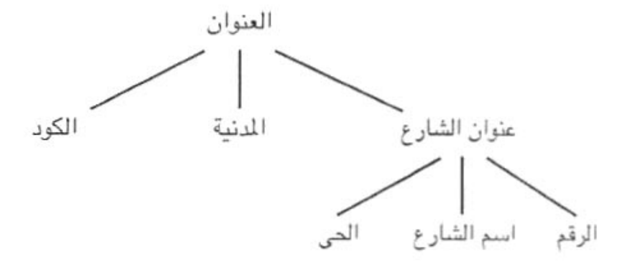
شكل رقم (٦-١٥) يوضح كينونتى E1 و C1 وقيم خصائصهما



(أ) الخصائص المركبة وذات القيمة الواحدة فقط:

بعض الخصائص يمكن تقسيمها إلى جزئيات صغرى ذات معنى مستقل بذاتها. على سبيل المثال: خاصية العنوان يمكن تقسيمها إلى خصائص أخرى كما هو مبين بالشكل رقم (٦-١٦).

شكل رقم (٦-١٦) هرمية الخصائص المركبة



ويطلق على الخاصية التى تتكون من عدة خصائص فرعية اسم الخاصية المركبة Composite ، بينما الخاصية غير القابلة للتقسيم تسمى بالخاصية البسيطة Simple ، وهى الخصائص ذات القيمة الواحدة فقط atomic. والخصائص المركبة يمكن أن تأخذ الشكل الهرمى كما وضح بالشكل رقم (٦-١٦). وقيمة الخاصية المركبة هى مجموعة قيم متلاصقة لخصائص القيمة الواحدة فقط. وتكون الخاصية المركبة مفيدة للمستخدم عندما يشار إليها كوحدة واحدة ، ولكن قد لا تكون ذات قيمة عندما يشار إلى أحد مكوناتها من دونها. ومن هنا يتبين متى يتم تقسيم الخاصية المركبة إلى خصائص ذات قيمة واحدة فقط أو تركها لحالها.

(ب) الخصائص الفردية والمتعددة القيم :

معظم الخصائص التى لها قيمة فردية تابعة لكينونة معينة ، يطلق عليها الخصائص فردية القيمة. على سبيل المثال: كينونة شخص معين لها قيمة فردية لخاصية العمر، فى حين أنه فى حالات أخرى قد تكون هناك خاصية لها مجموعة قيم لنفس الكينونة. على سبيل المثال: خاصية اللون لكينونة السيارة قد تكون : بيضاء حمراء - سوداء وغيرها. فمثل هذه الخصائص تسمى خصائص متعددة القيم. وكل كينونة تحتوى على خاصية لها قيم متعددة يكون لها حد أدنى وأعلى لعدد هذه القيم. كما هو موضح فى خاصية لون السيارة التى قد يتراوح عددها من قيمة واحدة إلى خمس قيم فقط.

(ج) الخصائص المستنتجة Derived Attributes :

فى بعض الحالات قد تكون هناك قيمتان أو أكثر لخاصية معينة مترابطتان. مثال ذلك: خاصيتا العمر وتاريخ الميلاد لكينونة شخص معين. بمعنى أخر يمكن أن يحدد عمر شخص معين من تاريخ اليوم وتاريخ ميلاده. ويطلق على خاصية العمر الخاصية المستنتجة. وبعض قيم الخاصية يستنتج من كينونات مترابطة ، على سبيل المثال: عدد الموظفين فى نوع كينونة "الإدارة" DEPARTMENT يمكن أن يحسب بعدد الموظفين الذين يعملون بالإدارة.

(د) الخصائص خالية القيمة Null :

فى بعض الحالات الخاصة بكينونة معينة قد لا تلحق قيمة لخاصية معينة ، فعلى سبيل المثال: فإن خاصية رقم الشقة فى العنوان تطبق فقط على عنوان العمارة كما فى المنازل العائلية الفردية وليس على رقم الشقة أو الغرفة. فى مثل هذه الحالات توضع قيمة خاصة تكون ملغاة. وتسمى هذه الخصائص خالية القيمة Null. وبناءً عليه فإن خاصية عنوان المنزل العائلى الفردى توضع به القيمة الملغاة Null.

نوع الكينونة ونطاق القيم وخصائص المفتاح :

(أ) نوع الكينونة:

تحتوى قواعد البيانات عادة على مجموعات من الكينونات تكون متشابهة.

مثال ذلك: مئات من موظفى شركة معينة قد يخزنون معلومات متشابهة تخص كل موظف. وتشترك كينونات الموظفين فى نفس الخصائص، ولكن كل كينونة قد يكون لها قيمتها الخاصة بها التى تخص كل خاصية على حدة. مثل هذه الكينونات المتشابهة تعرف بنوع الكينونة التى هى مجموعة من الكينونات لها نفس الخصائص المشتركة. ومعظم قواعد البيانات لديها أنواع كينونات كثيرة ، حيث إن كل نوع كينونة يتم توصيفه بواسطة اسم الكينونة وقائمة من الخصائص الخاصة بها. ويبين الشكل رقم (٦-١٧) نوعى كينونة أحدهما نوع كينونة "موظف" EMPLOYEE والآخر نوع كينونة "شركة" COMPANY وقائمة من الخصائص لكل منها. ويوضع فى كل نوع كينونة مجموعة من الكينونات القليلة لقيم تنتمى إلى خصائصها.

شكل رقم (٦-١٧) يوضح نوعى كينونة وبعض أعضاء الكينونات

Schema (Intention)	EMPLOYEE Name, Age, Salary	COMPANY Name, Place, Director
Instances (extension)	• e1 (Jamal, 47,$70000) • e2 (Asem,46$,75000) • e3 (Nader, 38,$60000)	• c1 (Plastic, Dammam, Ah-med) • c2 (Computer, Geddah, Al_Yaser)

يسمى توصيف نوع الكينونة بمخطط نوع الكينونة. وهو هيكل التوصيف الشائع لنوع كينونة معينة التى يشارك فيها أعضاؤها. ويصف المخطط اسم نوع الكينونة واسم ومعنى كل خاصية خاصة بها وأى قيود على أعضائها. ومجموعة أعضاء نوع الكينونة تسمى "الوقائع" Instances وقد يطلق عليها "امتداد" extension نوع الكينونة. ولا يتغير المخطط فى أغلب الأحيان لأنه يصف هيكل أعضاء نوع الكينونة فى حين أن الوقائع قد تتغير نتيجة الحذف والإضافة والتحديث.

(ب) نطاق القيم:

ترتبط كل خاصية بسيطة لنوع كينونة معينة بنطاق من القيم. هذا النطاق يصف مجموعة قيم قد تخصص لتلك الخاصية لكل كينونة على انفراد . وفى الشكل رقم (٦-١٧) لو كان مدى الأعمار المسموح بها لخاصية العمر Age بين (٧٠،١٦) لكل موظف Employee ، فإن وصف مجموعة القيم الخاصة لتلك الخاصية تكون مجموعة الأرقام الصحيحة Integer بين (٧٠،١٦). وأن مجموعة القيم التى تصف خاصية الاسم Name هى مجموعات من سلاسل الحروف الأبجدية (String) التى يفصل بين كل مجموعة حروف والأخرى مسافة blank وهكذا.

(ج) خصائص المفتاح:

يعتبر المفتاح الخاص بنوع كينونة معينة قيداً مهمًا على الكينونات المكونة لها (أعضائها). ولكل نوع كينونة عادة خاصية ذات قيم لا تتكرر مع الكينونات الأعضاء. ومثل هذه الخاصية تسمى خاصية المفتاح وقيمتها يتم استعمالها لتعريف كل كينونة بشكل لا يتكرر. وفى الشكل رقم (٦-١٧) تمثل خاصية الاسم Name المفتاح لنوع كينونة شركة Company؛ لأنها لا تسمح لشركتين أن يكون لهما نفس الاسم. وأحيانًا يتم تشكيل المفتاح من مجموعة من الخصائص معًا. بمعنى أن تجميع قيم هذه الخصائص لكل كينونة على حدة يجب أن يميزها عن غيرها. ومجموعة الخصائص التى يتم تجميعها لتشكيل المفتاح يمكن أن تكون مجمعة فى خاصية مركبة بحيث تصبح خاصية المفتاح لنوع الكينونة وإن كان هذا يتعارض مع القيود الخاصة بالنماذج العلاقية.

مثال (٦-٢):

يستكمل هذا المثال التوصيف المفاهيمى المبدئى لمثال شركة المشاريع الهندسية (الافتراضية) الوارد فى المثال (٦-٢). فطبقًا لقائمة المتطلبات الخاصة ببيانات الشركة يمكن توصيف أربعة أنواع من الكينونات[١]:

(١) نوع كينونة "الإدارة" DEPARTMENT وتشمل الخصائص التالية:

الاسم DName رقم الإدارة DNum ، الأماكن Location ، المدير Manager ، تاريخ بدء عمل المدير MDate. وتعتبر خاصية الأماكن Locations فقط خاصية متعددة القيم. ويمكن توصيف كل من خاصيتى الاسم DName ورقم الإدارة DNum كخصائص للمفتاح؛ لأنه من غير المتوقع أن يوجد إدارتان بهما نفس الاسم DName أو نفس الرقم DNum .

(٢) نوع كينونة "المشروع" PROJECT وتشمل الخصائص التالية:

اسم المشروع PName ورقم المشروع PNum والمكان PLocation والإدارة المشرفة CDept. وتشكل كل من خاصيتى الاسم PName والرقم PNum خصائص المفتاح .

(٣) نوع كينونة "الموظف" EMPLOYEE وتشمل الخصائص التالية:

اسم الموظف EName، رقم الضمان الاجتماعى SSN والجنس Sex، العنوان EAddr، المرتب Salary وتاريخ الميلاد BDate والإدارة التابع لها DName والمشرف Supervisor. وقد يكون كل من الاسم والعنوان الخصائص مركبة إلا إذا طلب المستفيد أن يكون الاسم مكونًا من محتويات فردية مثل الاسم الأول FName ، أو اسم الوسط MName أو الاسم الأخير LName وكذلك العنوان EAddr.

(٤) نوع كينونة "الإعالة" DEPENDENT وتشمل الخصائص التالية:

اسم الموظف EName ، اسم المعول DEName ، والجنس Sex ، تاريخ الميلاد BDate ، صلة القرابة REMP.

ومع ذلك لم يتم تمثيل حقيقة أن الموظف يمكن أن يعمل فى عدد من المشاريع ولا تمثيل لحقيقة عدد الساعات الأسبوعية التى يعملها الموظف فى كل مشروع. ويمكن

تمثيل هذه الحقيقة بخاصية مركبة متعددة القيم تسمى "يعمل على" Works-On ذات محتويات بسيطة هى: عدد الساعات الأسبوعية Hours وأسماء المشاريع التى يعمل بها Project. ويبين الشكل رقم (٦ – ١٨) مخططاً لكل نوع كينونة تم توصيفه والعديد من علاقات الربط الضمنية بين مختلف أنواع الكينونات. ففى حقيقة الأمر عندما تشير خاصية لنوع كينونة معينة إلى نوع كينونة آخر ، فلابد من وجود علاقة ربط. على سبيل المثال: خاصية المدير Manager فى نوع كينونة "الإدارة" DEPARTMENT تشير إلى الموظف الذى يدير الإدارة، وخاصية الإدارة المشرفة CDept لنوع كينونة "المشروع" PROJECT تشير إلى اسم الإدارة التى تشرف على المشروع. وخاصية المشرف Supervisor لنوع كينونة "الموظف" EMPLOYEE تشير إلى موظف آخر (الشخص الذى يشرف على هذا الموظف). خاصية الإدارة DName فى نوع كينونة "الموظف" تشير إلى الإدارة التى يعمل بها الموظف وهكذا. ويلاحظ فى الشكل رقم (٦–١٨) استخدام نوعين من الأقواس:

١. أقواس تبين الخصائص المتعددة القيم { } .

٢. أقواس تبين محتويات الخاصية المركبة ( ) .

شكل رقم (٦–١٨) التصميم المفاهيمى المبدئى لأنواع الكينونات الخاصة بشركة المشاريع الهندسية (الافتراضية)

DEPARTMENT

DName . DNum . { Location} . Manager . MDate

PROJECT

PName . PNum . Location . CDept

EMPLOYEE

EName (FName. MName . LName) . SSN . Sex . EAddr .
Salary . BDate . DName . Supervisor . { Works On (Project.Hour)}

DEPENDENT

EName . DEName . Sex . BDate . REMP

أنواع علاقات الربط Relationship types :

نوع علاقة الربط ووقائع علاقة الربط:

نوع علاقة الربط R بين n من أنواع كينونات (E_2 , E_1 .... E_1) هي مجموعة ارتباطات بين كينونات من هذه الأنواع. رياضياً: مجموعة وقائع علاقة الربط R هي ri حيث إن ri هي n من الكينونات الأعضاء ($e_1,e_2,...e_n$) وكل كينونة e_j في ri هي رقم نوع الكينونة Ej (حيث إن $1 \leq n\ j$)؛ لذا فان نوع علاقة الربط رياضياً هي نوع كينونة. وكل من أنواع الكينونات $E_1, E_2 ,...E_n$ تسهم Participate في نوع علاقة الربط R. وأيضاً كل من الكينونات الأعضاء e1,e2,...en تسهم في واقعة الربط ri.

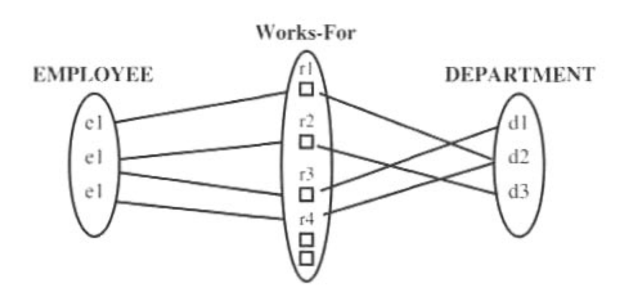
مثال (٦-٣):

تعتبر نوع علاقة الربط "يعمل في" Works-For بين نوعي كينونة "الموظف" EM-PLOYEE و"الإدارة" DEPARTMENT ، هي التي تربط كل موظف بالإدارة التي يعمل بها. كل واقعة في علاقة ربط "يعمل في" Works– For تربط كينونة موظف وكينونة إدارة، حيث يوضح الشكل رقم (٦-١٩) بعض الوقائع لعلاقة الربط "يعمل في" Works-For(٦).

درجة نوع علاقة الربط Arity of a Relationship type:

درجة نوع علاقة الربط هي عدد أنواع الكينونات التي تشترك في نوع علاقة الربط: ولذا فإن درجة علاقة الربط "يعمل في" Works-for هي علاقة ربط ثنائية Binary.

شكل رقم (٦-١٩) يبين بعض الوقائع لعلاقة الربط "يعمل في" Works-for



قيود على أنواع علاقات الربط:

أنواع علاقات الربط عادة لها قيود معينة تحدد إمكانيات تجميع الكينونات المشاركة فى وقائع علاقة الربط. ويتم تحديد هذه القيود من بيئة العالم الخارجى الذى يمثل علاقات الربط . على سبيل المثال ففى الشكل رقم (٦- ١٩) لو كانت هناك شركة معينة لها قاعدة تنص على أن الموظف يعمل فقط فى إدارة واحدة فقط عندئذ لا بد من وصف هذا القيد على المخطط. ويمكن التمييز بين نوعين أساسيين لقيود علاقات الربط التى تحدث بشكل متكرر نسبياً، هما[1]:

* **نسبة التعددية Cardinality ratio .**

* **المشاركة Participation .**

أولاً – قيد نسبة التعددية:

يصف هذا القيد عدد وقائع علاقات الربط التى يمكن للكينونة أن تشارك فيها. فعلى سبيل المثال: نسبة التعددية لنوع علاقة الربط الثنائية Works-for هى واحد – المتعدد (1:M)، وهذا يعنى أن كل كينونة بالقسم يمكن أن ترتبط بعدد من كينونات الموظف ولكن كينونة الموظف يمكن أن ترتبط بقسم واحد. ونسب التعددية الشائعة لأنواع علاقات الربط هى:

* واحد – لواحد (١:١)

* واحد – لمتعدد (I:M)

* متعدد – لمتعدد (M:N)

قيد المشاركة:

يتم توصيف هذا القيد فى حالة وجود كينونة مرتبطة بكينونة أخرى عن طريق نوع علاقة ربط معينة. ويوجد نوعان من قيود المشاركة هى :

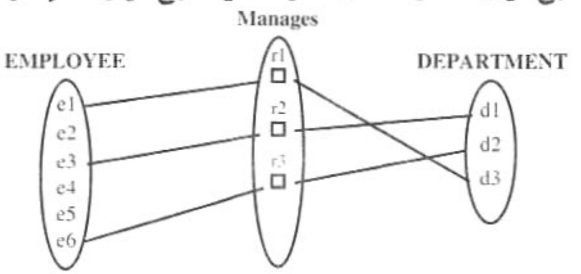
* قيود مشاركة كلية Total .

* قيود مشاركة جزئية Partial.

مثال (٦-٤):

لو فرض أن القواعد المنظمة لأعمال الشركة فى المثال (٦-٣) تبين أن كل موظف يجب أن يعمل فى إدارة واحدة ، عندئذ يمكن أن توجد كينونة الموظف فقط فى حالة مشاركتها فى وقائع علاقة الربط "يعمل فى" Works-For. فالمشاركة لنوع كينونة "الموظف" EMPLOYEE فى علاقة الربط Works-For تسمى قيد مشاركة كلية . وهذا يعنى أن كل كينونة لكينونات الموظف يجب أن ترتبط بكينونة إدارة عن طريق نوع علاقة الربط "يعمل فى" .Works-For وأحيانًا يسمى قيد المشاركة الكلية بالتبعية الوجودية .existence dependency بينما فى علاقة الربط "يدير" Manages الموضحة فى الشكل رقم (٦-٢٠) يتضح أنه من غير المتوقع أن كل موظف يدير إدارة ، لذلك فإن مشاركة نوع كينونة "الموظف" فى نوع علاقة الربط Manages هى مشاركة جزئية. وهذا يعنى أن بعض أو جزء من مجموعة كينونات الموظفين ترتبط بكينونة إدارة عن طريق نوع علاقة الربط "يدير" Manages وليس الكل. وتجدر الإشارة إلى أن قيود نسبة التعددية والمشاركة معًا تعتبر قيودًا هيكلية Structure Constrains لنوع علاقة الربط.

الشكل رقم (٦-٢٠) يبين نوع علاقة الربط "يدير" Manages (١:١) بمشاركة جزئية لنوع كينونة "الموظف" ومشاركة كلية لنوع كينونة "الإدارة"

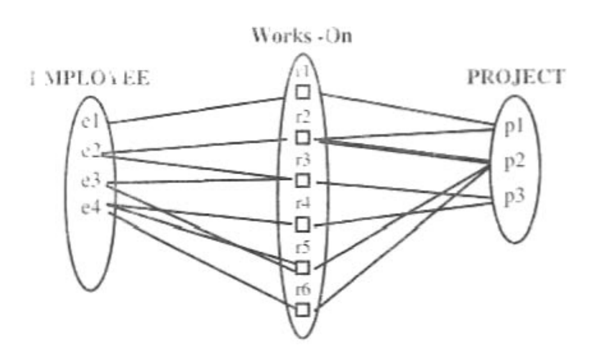


خصائص أنواع علاقات الربط:

إن أنواع علاقات الربط لها أيضًا خصائص تشابه خصائص أنواع الكينونات. مثال ذلك : عندما تسجل عدد ساعات العمل hours فى الأسبوع للموظف الذى "يعمل على" Works-On مشروع معين ، فإن هذه الساعات يمكن أن تحفظ فى واقعة علاقة

الربط "يعمل على" .Works-on: لذا لا بد أن تشمل علاقة الربط "يعمل على" Works-on على خاصية عدد ساعات العمل hours. كما هو فى الشكل رقم (٦-٢١).

شكل رقم (٦-٢١) يبين نوع علاقة الربط "يعمل على" Works-on متعدد-لمتعدد (M:N)



ويمكن أيضًا أن تشمل علاقة الربط "يدير" Manages فى الشكل رقم (٦-٢٠) على خاصية تاريخ بدء عمل المدير Mdate. ويلاحظ أن خصائص أنواع علاقات الربط (1:N) و (1:1) يمكن أن تتضمن كخصائص لأحد أنواع الكينونات المشاركة. مثال ذلك: خاصية تاريخ بدء عمل المدير MDate يمكن أن تكون كخاصية إما فى نوع كينونة "الموظف" أو نوع كينونة الإدارة". ومع ذلك تنتمى إلى نوع علاقة الربط "يدير" Manages. وذلك لأن نوع العلاقة الربط "يدير" Manages، هى واحد - لواحد (1:1): وذلك لأن كل نوع كينونة "الإدارة" أو نوع كينونة "الموظف" تشارك على الأكثر فى واقعة واحدة لعلاقة الربط. أما بخصوص نوع علاقة الربط واحد - لمتعدد (N: 1) فإن خاصية الملحقة بعلاقة الربط يمكن أن تضمن فقط لخاصية فى نوع الكينونة المشاركة على جانب مناسب لها فى نوع علاقة الربط. فعلى سبيل المثال: فى علاقة الربط "يعمل فى" Works-for فإن خاصية تاريخ الموظف "بدء تاريخ تعيين الموظف EDate" فى نوع كينونة "الإدارة" هى خاصية مضمنة فى نوع الكينونة "الموظف": لأن علاقة الربط هى واحد-لمتعدد (1:N). كذلك كينونة كل موظف تشارك على الأكثر فى واقعة واحدة لعلاقة الربط "يعمل فى" Works-for: لذا فإن مشاركة كينونة الموظف هى التى تحدد قيمة تاريخ بدء عمل الموظف EDate.

أنواع الكينونات الضعيفة Weak Entity type :

بعض أنواع الكينونات قد لا يكون لها خصائص المفتاح ، وهذا يتضمن عدم القدرة على التمييز بين بعض هذه الكينونات : لأن تجميع قيم خصائصها يمكن أن يكون متطابق. مثل هذه الأنواع تسمى أنواع الكينونات الضعيفة. والكينونات التى تنتمى إلى نوع الكينونة الضعيفة يتم تعريفها بكونها مرتبطة بكينونة محددة من نوع كينونة أخرى. ويسمى نوع الكينونة المحددة للكينونة الضعيفة باسم **صاحب التعريف** identifying owner ويسمى نوع علاقة الربط المرتبط بنوع الكينونة الضعيفة باسم **علاقة ربط التعريف** identifying relationships لنوع الكينونة الضعيفة. ودائماً نوع الكينونة الضعيفة له قيد المشاركة الكلية (تبعية الوجود). ومع ذلك ليس كل نتائج تبعية الوجود فى نوع الكينونة الضعيفة. مثال ذلك : نوع كينونة "الإعالة" DEPENDENT المرتبطة بنوع كينونة "الموظف" والتى تحفظ الذين يتم إعالتهم من قبل الموظف عبر علاقة الربط واحد - لمتعدد (1:N) تعتبر نوع كينونة ضعيفة. حيث يمكن أن يكون اثنان من المعالين من قبل موظفين مستقلين لهما نفس قيم الخصائص الخاصة بهما ومع ذلك لا يزالان كينونات مميزة.

نوع الكينونة الضعيف له مفتاح جزئى يتكون من مجموعة خصائص تكون وحيدة تستطيع أن تعرف الكينونات الضعيفة المرتبطة مع نفس الكينونة الأب Owner. وأحياناً أخرى تمثل أنواع الكينونات الضعيفة بخصائص متعددة القيم المركبة [١]،[٦].

مثال (٦-٥) :

يمكن فى هذا المثال عمل تكرير لتصميم كينونة - علاقة ER فى قاعدة بيانات شركة المشاريع الهندسية (الافتراضية) فى المثالين السابقين، وذلك بتغيير الخصائص التى تمثل علاقات الربط فى أنواع علاقات الربط. وقيود نسبة التعددية والمشاركة لكل نوع علاقة ربط تكون محددة من خلال قائمة المتطلبات السابقة. وفى مثال الشركة يمكن تحديد أنواع علاقات الربط التالية[١]:

(١) نوع علاقة الربط "يدير" Manages وهى واحد - لواحد (١:١) بين نوعى كينونة "الموظف" EMPLOYEE و"الإدارة" DEPARTMENT حيث إن مشاركة الموظف

جزئية ، أما مشاركة الإدارة ليست واضحة. ويفترض ضمنيًا هنا أن المشاركة كلية بحيث إن مدير الإدارة يظل بها كل الوقت. وتخصص لعلاقة الربط "يدير" خاصية تاريخ بدء عمل المدير MDate.

(٢) نوع علاقة الربط "يعمل فى" Works-For هى واحد - لمتعدد (1:N) بين نوعى كينونة "الإدارة" DEPARTMENT و"الموظف" .EMPLOYEE وكليهما ذات مشاركة كلية.

(٣) نوع علاقة الربط "يراقب" Controls هى واحد - لمتعدد (1:N) بين نوعى كينونة "الإدارة" DEPARTMENT و"المشروع" .PROJECT ومشاركة "المشروع" PROJECT كلية.

(٤) نوع علاقة الربط "الإشراف" Supervision الأحادية هى واحد - لمتعدد (1:N) بين الموظف (فى وظيفة المشرف) والموظف (فى وظيفة المشرف علية). وتكون مشاركة جزئية لكليهما؛ لأنه ليس كل موظف له مشرف وليس كل موظف مشرفًا.

(٥) نوع علاقة الربط "يعمل على" Works_On هى متعدد - لمتعدد (N:M). ويخصص لها خاصية عدد الساعات hours ، حيث إن المشروع يعمل به عدد من الموظفين. المشاركة كلية لكل من نوعى الكينونة.

(٦) نوع علاقة الربط "يعول" Dependents هى واحد - لمتعدد (1:N) بين نوعى كينونة "الموظف" EMPLOYEE "الإعالة" DEPENDENT (نوع كينونة "الإعالة" ضعيف). ومشاركة الموظف جزئية ، فى حين أن مشاركة نوع كينونة "الإعالة" كلية .

يبين الشكل رقم (٦-٢٢) نموذج كينونة - علاقة ربط لمخطط قاعدة بيانات شركة المشاريع الهندسية (الافتراضية) حيث إن:

(١) أنواع كينونات "المشروع" PROJECT و "الموظف" EMPLOYEE والإدارة DE-PARTMENT تكتب داخل أشكال مستطيلة.

(٢) أنواع علاقات الربط مثل "يعمل على" Works-On و"يدير" Manages و"يعمل فى" Works-For و "يراقب" Controls تكتب داخل أشكال المعين بحيث ترتبط بأنواع الكينونات المشاركة باستخدام خطوط مستقيمة.

(٣) الخصائص تكتب داخل أشكال بيضاوية، وكل خاصية تلحق بنوع كينونتها أو بنوع علاقة ربط معينة بواسطة خط مستقيم.

(٤) الخصائص المتعددة القيم يتم تمثيلها باستخدام الشكل البيضاوى المزدوج مثل خاصية الأماكن Location المخصصة لنوع كينونة "إدارة" DEPARTMENT.

الخصائص المركبة مثل خاصية الاسم Name فى نوع كينونة "الموظف" تلحق محتوياتها بالشكل البيضاوى الخاص بالخاصية. أما عن خاصية المفتاح فيكتب تحتها خط. أما الخصائص المستنتجة يتم تمثيلها بشكل بيضاوى فقط.

(٥) أنواع الكينونات الضعيفة يتم تمييزها بوضعها فى مستطيل مزدوج، وعلاقات الربط التى ترتبط بها تكتب فى شكل معين مزدوج، كما فى نوع الكينونة "الإعالة" DEPENDENT، ونوع علاقة الربط "يعول" Dependents. والمفتاح الجزئى لنوع الكينونة يكتب تحته خط منقوط.

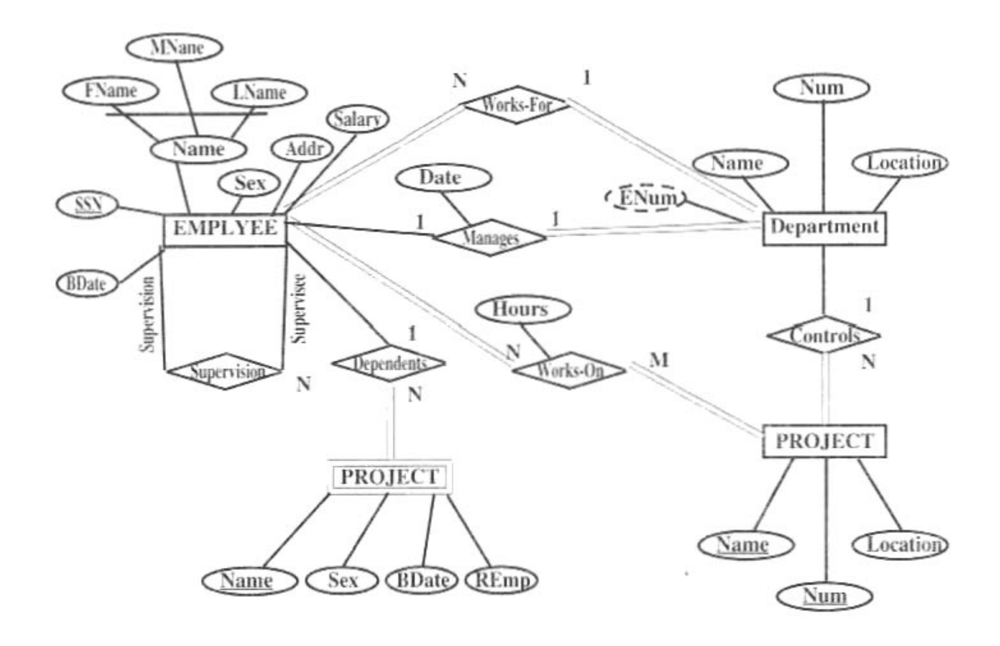
(٦) النسب التعددية لأنواع علاقات الربط الثنائية يتم توصيفها بإلحاق رموز التعددية 1.M.N على كل وصلة مشاركة.

(٧) قيد المشاركة يتم توصيفه بواسطة خط فردى للمشاركة الجزئية وبخط مزدوج للمشاركة الكلية.

(٨) فى نوع علاقة الربط "الإشراف" Supervision الأحادية تكتب أسماء الوظيفة على الجانبين: لأن نوع كينونة "الموظف" تقوم بكل من الوظيفتين فى تلك العلاقة.

(٩) يمكن استخدام تدوينات بديلة لإظهار القيود الهيكلية على أنواع علاقات الربط بعرض قيمتين صحيحتين (min . max) على كل خط يمثل المشاركة لنوع كينونة فى نوع علاقة ربط معينة.

شكل رقم (٦ - ٢٢) نموذج كينونة - علاقة لقاعدة بيانات شركة

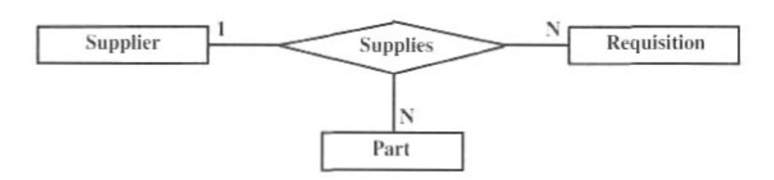


تفكيك علاقات الربط :

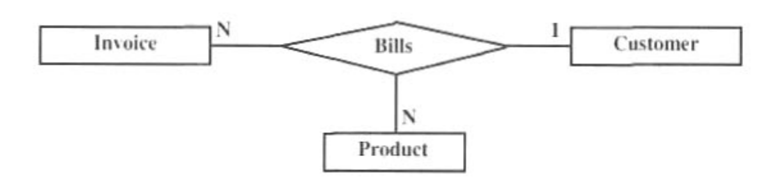
أحد الاعتراضات المعنوية العديدة فى استعمال نماذج كينونة - علاقة ER هو إما أن يقدر استعمال علاقة ربط وحيدة بين العديد من الكينونات أو مجموعة علاقات ربط أبسط لتمثيل ارتباطات معقدة. ويوفر نموذج كينونة - علاقة ER تدوينًا يشكل استقلالية للأسلوب الفنى: مما يؤدى إلى فهمه بواسطة كل من محلل البيانات والمستفيد النهائى للنظام. ولكن العديد من أصحاب المهن يصادفون صعوبات فى فهم علاقات الربط التى تشمل ثلاثة أنواع كينونات. إحدى المشكلات الكثيرة التكرار التى يتم تحديدها بواسطة محلل البيانات أثناء عملية نمذجة البيانات هى: نمذجة علاقة الربط التى يعتقد أنها ثلاثية مع عدم التأكد من ذلك؛ مما يؤدى إلى طرح السؤال التالى: كيف يمكن تحديد علاقات ربط ثنائية أبسط وكافية؟

نفس عدم التأكيدات في وقائع نادرة نسبيًا عندما يتم الأخذ في الاعتبار علاقات ربط ذات الدرجة الرابعة أو أكثر. لو أخذ في الاعتبار كمثال نموذج كينونة-علاقة ER في الشكل (٦-٢٣)، علاقة الربط "يورد" Supplies في الشكل رقم (٦-٢٣) تعرف مطلباً لتسجيل القطعة Part التي تظهر في كل طلب requistion بالإضافة إلى المورد الذي يورد كل جزء. وبالمثل في الشكل رقم (٦-٢٤) تعرف علاقة الربط "يقدم فاتورة" bills مطلبًا لتسجيل المنتجات Products التي تظهر على كل فاتورة Invoice بالإضافة إلى العميل Customer بكل فاتورة.

شكل رقم (٦-٢٣) يوضح علاقة الربط "يورد" الثلاثية



شكل رقم (٦-٢٤) يوضح علاقة الربط "يقدم فاتورة" الثلاثية



كلتا علاقتي الربط يتشابه في قيود التعددية حيث إن قيود التعددية لكل منهما N-N-1. ومثل هذه القيود تحدد التجميعات المسموح بها لمشاركة أنواع الكينونات. على سبيل المثال : القيمة (١) لنوع كينونة "المورد" Supplier في الشكل (٦-٢٣) تشير إلى القطعة Part الموجودة على الطلب requisition المقدم، والتي يمكن توريدها عن طريق مورد واحد على الأكثر. أما القيمة N لنوع كينونة "الطلب" REQUISITION تشير إلى أن المورد المنوط بتوريد القطعة المطلوبة ، يمكن أن يوردها على أرقام غير محددة للطلبات.

وعلى الرغم من التشابه الظاهر للمثالين، إلا أن علاقات الربط تمثل متطلبات بيانات مختلفة معنوياً. وأحد الأساليب لتقدير الفارق بشكل بديهى، هو فحص عينة من وقائع علاقتى الربط. وواقعة علاقة الربط يمكن التفكير فيها كارتباط بين واقعة واحدة لكل من أنوا ع الكينونات المشاركة.

واقعة علاقة الربط "يقدم فاتورة" bills تتكون من منتج Product محدد مسدد ثمنه من قبل عميل محدد كجزء من فاتورة محددة. ويوفر الشكل رقم (٦-٢٥) تمثيلاً لبعض الوقائع بواقعة علاقة الربط لكل صنف. على سبيل المثال: الصف الأول يشير إلى رقم المنتج (١٠٠) المسدد.

شكل رقم (٦-٢٥) يوضح تمثيل فئة الوقائع لعلاقة الربط "يقدم فاتورة"

Invoice #	Customer #	Product #
10001	1111	100
10001	1111	200
10001	1111	300
10002	2222	200
10002	2222	400
10003	1111	100
10003	1111	200

ثمنه من قبل العميل رقم (١١١١) كجزء من الفاتورة رقم (١٠٠٠١). وبناء على ذلك، لا توجد مجموعة من علاقات الربط الثنائية تنقل نفس مجموعة الحقائق مثل علاقة الربط الثلاثية.

على سبيل المثال: الارتباطات المستخلصة من شكل رقم (٦-٢٦) بين الطلب Requisition والقطعة Part، بالإضافة إلى الطلب والمورد، فى شكل رقم (٦-٢٧) توضح فقدان تركيز أى قطعة لرقم الطلب (١٠٠٠) التى يتم توريدها بواسطة المورد رقم (١٠) وذلك عكس التى يتم توريدها بواسطة المورد رقم (٢٠). وذلك يعنى أن أى تفكيك آخر لنواتج علاقات ربط ثنائية بنفس التشابه يؤدى إلى فقدان البيانات. وبناء عليه، فإن

علاقة الربط "يورد" Supplies فى الشكل رقم (٦-٢٣) يجب أن تبقى ثلاثية فى حين أن علاقة الربط "يقدم فاتورة" bills يمكن أن تفكك إلى علاقتى ربط ثنائية.

شكل رقم (٦-٢٦) يوضح تمثيل فئة الوقائع لعلاقة الربط "يورد"

Requidition #	Part #	Suppliert #
1000	200	10
1000	300	10
1000	500	20
1001	200	10
1001	500	10

شكل رقم (٦-٢٧) جداول علاقية غير ملائمة لبيانات "الطلب"

شكل رقم (٦-٢٧أ) جدول "الطلب" يتضمن "رقم القطعة"

Requidition #	Part #
1000	200
1000	300
1000	500
1001	200
1001	500

شكل رقم (٦-٢٧ب) جدول "الطلب" يتضمن "رقم المورد"

Requidition #	Suppliert #
1000	10
1000	20
1001	10

متى وكيف ينبغى تفكيك علاقات الربط فى نموذج كينونة – علاقة ER؟ باستخدام طريقة **الخطوات الثمان** التى ترسم بشكل صعب على التشابهات، بين تقليل جداول قاعدة البيانات العلاقية وعلاقات ربط نموذج كينونة – علاقة ER.

قيود تعددية علاقة الربط Cardinality Constraints :

قيمة (N) على وظيفة "القطعة" Part لعلاقة الربط "يورد" تعنى أن واقعة الطلب المزدوجة مع واقعة المورد ليس لها حد أقصى على الرقم المرتبط بوقائع القطعة. بمعنى آخر فإن المورد المذكور يمكن أن يورد العديد من القطع مختلفة على الطلب المحدد ويتم توصيفها كالآتى:

Cmax = (SR.P) N.

أما القيد Cmax (SR,S) = 1 فيعنى أن أى مجموعة من رقم الطلب مع رقم القطعة يرتبط على الأكثر برقم مورد واحد. أى أن القيد Cmax(a,b)=1 له نفس المعنى كما فى التبعية الوظيفية:

$$FD: a \rightarrow b$$

جداول التعددية Cardinality Tables :

فى الشكل رقم (٦-٢٣) ، (٦-٢٤) القيود المعرفة غير كافية لاتخاذ قرار بما إذا كانت علاقات الربط تفكك أم لا، توصيفاً يتضمن (١٢) قيد تعددية كل علاقة ربط ثلاثية أفضل من ثلاثة قيود، قيود Cmax للفئة الثانية عشرة (١٢) الناتجة يمكن أن توصف باستعمال جدول التعددية كما هو مبين فى الشكل رقم (٦-٢٨) لعلاقة الربط "يورد" . Supplies

إن كل صف فى جدول التعددية يمثل مجموعة وظائف محددة للوظيفة a ، ويوجد صف واحد لكل وظيفة a محتملة. بالمثل فإن الأعمدة تمثل كل الفئات الممكنة لوظيفة b. وهكذا كل خلية فى الجدول تمثل قيد Cmax(a,b) فردى. والخلايا المتداخلة تكون مظللة ولا تستعمل. مثال ذلك: القيمة فى الصف الأول "المورد" Supplier العمود الثانى "الطلب" requisition للشكل رقم (٦-٢٨) تشير إلى القيد Cmax (S,R) = N لعلاقة الربط "يورد" Supplies. بمعنى آخر فإن المورد المحدد يمكن أن يكون مرتبطاً بعدد

أقصى غير محدد للطلبات فى سياق علاقة الربط "يورد". يوضح شكل رقم (٦-٢٨) الاتفاقات المقترحة التالية لاستعمال جداول التعددية:

(أ) كل جدول يقسم إلى أجزاء sections مبنية على عدد الوظائف فى كل من a . b. فعلى سبيل المثال: الجزء الأيمن الأعلى للشكل رقم (٦-٢٨) يتضمن كل قيد Cmax (a.b) ، حيث إن (a) لها وظيفة واحدة و (b) لها وظيفتان. ويشار إلى هذا الجزء بالجدول 1x2. أما فى أقصى اليسار من نفس الشكل فتوجد الأجزاء 1x1 ، 2x1 على التوالى. فى حين أن الأجزاء التى لها كل الخلايا مظللة ينبغى أن تحذف.

(ب) يعنون كل صف للجزء 1x1 للجدول بكينونة واسم وظيفة لتعرف وظيفة فردية بشكل لا يتكرر. وتكون أسماء الوظائف ضرورية فى حالة مشاركة الكينونة فى العديد من الوظائف. كل الأعمدة والصفوف الأخرى تكون معنونة باختصارات أسماء أنواع الكينونات الوظائف. تم توصيف ثلاثة قيود تعددية لعلاقة الربط "يورد" Supplies فى الشكل رقم (٦-٢٣) تظهر فى الجزء 2x1 للجدول فى شكل رقم (٦-٢٨) تعطى نفس المعنى بالضبط. ويختلف حجم جدول التعددية اعتماداً على عدد الوظائف فى علاقة الربط.

ويبين الشكل رقم (٦-٢٨) تنظيم جدول التعددية لعلاقة الربط من الدرجة الرابعة 4-ary. واكتمال مثل هذا الجدول ليس نموذجاً مثبطاً للهمة؛ لأن معظم المدخلات entries تتجه إلى أن تكون N ، ومن ثم يمكن توصيفها بقيود معنوية قليلة فقط. وغالباً تكون هناك حاجة لنمذجى البيانات بشكل نادر فى حالة التعامل مع علاقات الربط التى تشمل أكثر من ثلاثة وظائف. وأن اكتمال جدول التعددية لعلاقة الربط وليكن للدرجة N ، تساعد محللى البيانات لعمل تحليل أكثر اكتمالاً لطبيعة علاقة الربط.

شكل رقم (٦ - ٢٨) جدول التعددية لعلاقة الربط "يورد"

	S	P	P
Supplier		N	N
Requistion	N		N
Part	N	N	
SR		N	N
SP	N	N	N
RP	1	N	

	SR	SP	RP
Supplier		N	N
Requistion	N	N	N
Part	N	N	

قواعد التناسق Consistency :

يوجد القليل من قواعد الاتساق التى ينبغى أن تتبع فى حالة اكتمال جدول التعددية. فعلى سبيل المثال: عندما يكون القيد فى الشكل رقم (٦-٢٩) هو Cmax (I,C) ، عندئذ فإن القيد Cmax (IP,C) ينبغى أن يساوى 1. وفى حالة وجود عميل Customer واحد فقط له الفاتورة المحددة Invoice ، وعندئذ يمكن أن يوجد فقط عميل واحد للمنتج المذكور على الفاتورة. أى قيمة تعادل 1 فى الجزء 1 × 1 فى جدول التعددية الثلاثى تصير 1 فى نفس العمود فى الجزء 1 × 2. وهذا الحصر هو تطبيقًا لقاعدة الاكتمال Augmentation لتأكيد الاتساق فى جدول التعددية لعلاقة الربط ذات الدرجة الثلاثية. ويمكن تعميم مسلمات أرمسترونج Armstrong's axioms المستعملة فى تصميم قواعد البيانات العلاقية وتطبيق هذه القواعد على القيد Cmax للقيمة N أو لأى قيمة صحيحة.

شكل رقم (٦-٢٩) جدول التعددية لعلاقة الربط "يقدم فاتورة"

	I	C	P
Invoice		1	N
Customer	N		N
Product	N	N	

	IC	IP	CP
Invoice		N	N
Customer	N	N	N
Product	N	N	

IC		N	N
IP	N	1	N
CP	N	N	

تصميم قاعدة البيانات العلاقية :

* تبنى نظرية تصميم قاعدة البيانات العلاقية على مفاهيم المفاتيح والتبعيات الوظيفية لتعريف هذا النوع لتفكيك الجدول العلاقى. فعلى سبيل المثال : فى الجدول (1) التبعية الوظيفية :

$$F : I \rightarrow C$$

تعنى أن أى فاتورة Invoice يمكن أن ترتبط على الأكثر بعميل واحد فى الجدول، والتى تكافئ :

$$Cmax (I,C) = 1.$$

نموذج كينونة - علاقة EP :

أحد المراجع الأولية لتفكيك علاقات ربط كينونة - علاقة يظهر فى سياق طريقة تحويل نموذج كينونة - علاقة إلى تعريفات قواعد البيانات العلاقية. وتتضمن الطريقة الشكل الطبيعى لنماذج كينونة - علاقة التى تستعمل لتؤكد أن نتائج نموذج كينونة- علاقة في جداول علاقية في الشكل الطبيعي الثالث أو الرابع أو الخامس بمجرد أن يتم تحويل النموذج. ونموذج كينونة - علاقة الذى له هذه الخاصية يقال انه يحقق "الشكل الطبيعى". وهناك شروط عديدة يتم توفيرها لاختيار ما إذا كانت علاقة الربط لنموذج كينونة - علاقة تحقق الشكل الطبيعى. مثال ذلك: لمثل هذا الشرط لو كانت التبعية الوظيفة:

$$F : a \rightarrow b$$

تحقق علاقة الربط ، وأن a هى مجموعة الكينونات المشاركة ، و b كينونة مشاركة واحدة ، ولا تمثل a مفتاح علاقة الربط، فإن علاقة الربط عندئذ لا تحقق الشكل الطبيعى. وهذا الشرط يمكن تطبيقه على التبعية الوظيفية :

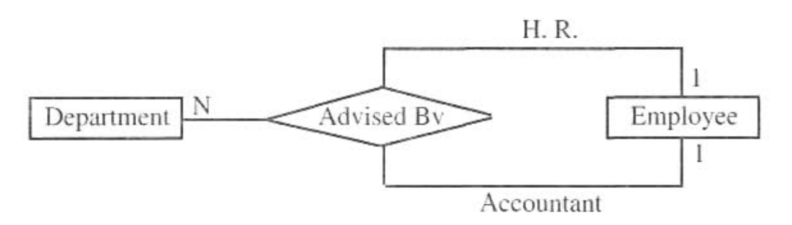
$$F : I \rightarrow C$$

لعلاقة الربط "يقدم فاتورة" Bills ، حيث إن مفتاح هذه العلاقة يتضمن كلاً من الفاتورة Invoice والمنتج Product: لذلك فإن علاقة الربط "يقدم فاتورة" ليست فى الشكل الطبيعى.

معيار تفكيك علاقات نموذج كينونة - علاقة يتم توفيرها كجزء لطريقة أخرى لتحويل نماذج كينونة - علاقة إلى تعاريف قواعد البيانات العلاقية. ويصف هذا المعيار أن علاقة الربط ذات الدرجة الثلاثية يمكن أن تفكك إلى علاقات ثنائية إذا - وإذا فقط - انتهك التمثيل العلاقى المكافئ لعلاقة الربط الثلاثية الشكل الطبيعى الرابع 4 NF .

وفى علاقة الربط الثلاثية فى الشكل رقم (٦-٣٠) ، كل ادارة يتم تخصيصها لمشرف واحد للموارد البشرية، ومشرف واحد للمحاسبة. حيث إن مشرفى الموارد البشرية والمحاسبية يشرفون على العديد من الإدارات.

شكل رقم (٦-٣٠) كل إدارة يتم الإشراف عليها من قبل مشرف الموارد البشرية (.H. R) ومحاسب



والنموذج العلاقى المكافئ يتضمن جدولاً علاقيًا واحدًا لكل من كينونات الادارة والموظفين ، بالاضافة إلى جدول علاقى منفصل لعلاقة الربط "يشرف" كالآتى:

DEPT (DEPT #, Dname , Dphone....)

EMP (EMP#,Ename,EBDate,Ephone)

ADVISES (DEPT#,HR-OFFICER-EMP#,Accountant-EMP#)

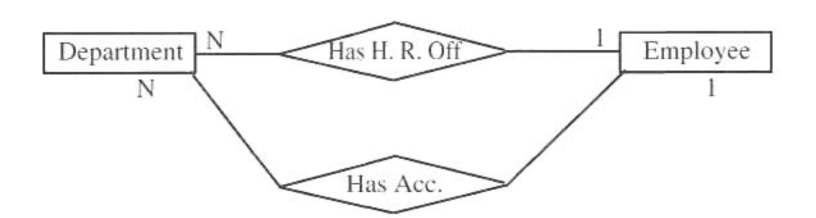
حيث إن هذه المخططات العلاقية فى الشكل الطبيعى الخامس NF٥ ، فإن معيار التفكيك يشير إلى أن علاقة الربط "يشرف" لا ينبغى أن تفكك. ومع ذلك استعمال هذه العلاقة الثلاثية يتضمن بشكل غير صحيح علاقات ربط ثنائية غير كافية. وهوية مشرف الموارد البشرية والمحاسبية مستقلة؛ ومن ثم يمكن تمثيلها بعلاقات ثنائية كما فى الشكل رقم (٦-٣١). والتحويل المبنى على نتائج الشكل رقم (٦-٣١) فى المخططات العلاقية المختلفة التالية ، وهى أيضًا فى الشكل الطبيعى الخامس 5NF.

DEPT (DEPT# , Dname , Dphone , HR-EMP#, ACCOUNTANT-EMP#)

EMP(EMP#,Ename,EBDate , Ephone)

وهذا يبين أن بعض علاقات نموذج كينونة - علاقة ER يمكن أن يتم تقسيمها حتى عندما تكافئ جداول علاقية فى الشكل الطبيعى الخامس 5NF. وهكذا يتضح عدم دقة معيار التفكيك السابق.

شكل رقم (٦-٣١) يوضح علاقتى ربط ثنائية لمشرفى الإدارة



ربما يكمن معظم اللبس فى معالجة تفكيك علاقة نموذج كينونة - علاقة فى تحليل قيود التعددية لمحتوى المفتاح. ومصطلح "التعددية الثلاثية" يستعمل للإشارة إلى نوع القيد الذى يظهر بشكل طبيعى فى نموذج كينونةعلاقة كما فى شكل رقم (٦-٢٤). مثل هذه القيود تناظر مدخلات فى الجزء 2×1 لجدول التعددية. التعدديات التى تقيد علاقات الربط الثنائية دلاليًا يتم تحليلها لتناظر المدخلات فى الجزء 1x1 لجدول التعددية.

طريقة الخطوات الثمان :

هذه الطريقة مصممة لاتخاذ قرار تفكيك علاقة الربط R الفردية ذات الدرجة N n-ary[٦]. ولسهولة التوضيح سيتم التركيز على علاقة الربط الثلاثية. وكيفية إتمام هذا التفكيك. وبمجرد الانتهاء من تحليل علاقة الربط الأصلية R إلى علاقات الربط n-ary جديدة. أيضًا يمكن أن يتم تحليل تلك العلاقات بنفس الخطوات الثمان ، كل منها بشكل منفصل عن الآخر. والخطوات الثمان فهذه الطريقة هى كالآتى:

(١) أكمل جدول التعددية لعلاقة الربط R. كل قيد Cmax = 1 حيثY a b r ⊂ يمثل تفكيك محتمل لعلاقة الربط إلى علاقتى ربط هما R1 ، R2. حيث تمثل r فئة كل وظائف علاقة الربط R. ويلاحظ أن لعلاقة الربط الثلاثيةR. r ⊂ a b لهذه القيود فقط Cmax (a.b) فى الجزء 1×1 لجدول التعددية. علاقة الربط R1 تتضمن الوظائف فى a b وعلاقة الربط فى R2 تتضمن الوظائف فى r - b. لو أن تفكيكًا واحدًا على الأقل يحتمل أن يتم تعريفة ، ينبغى اختيار إحدى الخطوات من ١.١ إلى 1.3 .

١/١ لكل تفكيك محتمل:

(أ) ينبغي استخلاص قيود التعددية لعلاقتي الربط R1,R2 من جدول التعددية لعلاقة الربط R. وفي حالة تضمين العلاقة المفككة أكثر من وظيفتين، ينبغي إنشاء جدول تعددية لتلك العلاقة.

(ب) ينبغي مقارنة مجموعة قيود التعددية المحددة لعلاقة الربط R بمجموعة القيود المحمية بواسطة النموذج المفكك. حيث تتضمن المجموعة المحمية كل القيود المحددة لعلاقتي الربط R1,R2 ، بالإضافة إلى القيود التي يمكن استنتاجها عن طريق استعمال قواعد الاتساق.

٢/١- في حالة وجود واحد أو أكثر من التفكيكات المحتملة التي تحمي كل القيود ، فعندئذ يمكن اختيار أي من هذه التفكيكات.

٣/١- في حالة عدم حماية التفكيك المحتمل لكل القيود ، عندئذ ينبغي اختيار واحد من هذه التفكيكات التي تحمي العدد الأكبر من القيود.

(٢) التفكيك المبني على التبعيات متعددة القيم:

في حالة وجود علاقة الربط R بدون تفكيك في الخطوة 1 ، فإنه ينبغي أن يحدد ما إذا كانت a متعددة القيم ، أي يتم تضمينها a _ b . c.

أولاً: أنه ينبغي تحديد الفئة المحتملة للقيم المتعددة على أن يأخذ في الحسبان علاقة الربط R. ولتعريف هذه الفئة ينبغي تعريف كل التغييرات الأساسية الممكنة في عدد الوظائف n لعلاقة الربط R في ثلاثة فئات أو وظائف هي a . b . c حيث إن r = a b c. فعلى سبيل المثال: يوجد ثلاث فئات متعددة لقيم محتملة ينبغي أن تأخذ في الحسبان لعلاقة الربط الثلاثية R هي:

$$R : A \to\to B|C, \quad B \to\to A|C, \quad C \to\to A|B.$$

ويلاحظ أن علاقة الربط الرباعية (ذات الوظائف الأربع) ، يوجد لها اثنتا عشرة قيمة متعددة محتملة للتبعية $A \to\to B|CD$ ، وست قيم متعددة محتملة للتبعية $AB \to\to C|D$. وبهذا يصل إجمالي القيم المتعددة المحتملة ثماني عشرة قيمة.

ثانيًا : تحليل كل احتمال متعدد القيم للتبعية a---b|c كالآتى (بافتراض أن علاقة الربط R ليست مفككة بالخطوة ١):

Cmax (a , c) > 1 , Cmax (a , b) > 1

وهذا يعنى أن علاقتى الربط الفرعتين متعددتا القيم. ولتحديد ما إذا كانت العلاقتان الفرعتين مستقلتين ، يطرح السؤال التالى:

"هل توجد معنويًا أى وقائع للوظيفة b مرتبطة بأى وقائع للوظيفة C بواسطة علاقة الربط R ؟"

لو كانت الإجابة كانت "لا" ، فإن التبعية a→→b|c تكون متعددة القيم؛ لذا يجب فك علاقة الربط إلى علاقتى ربط R1,R2. حيث إن علاقة الربط R1 تتضمن الوظائف فى ab وعلاقة الربط R2 تتضمن الوظائف فى ac.

(٣) التفكيك إلى ثلاث علاقات ربط أو أكثر:

فى حالة عدم تفكيك علاقة الربط فى الخطوة ١ أو ٢ ، فأنه ينبغى تحديد ما اذا كانت علاقة الربط R يمكن أن تفكك إلى ثلاث علاقات ربط أو أكثر مبنية على انتهاك الشكل الطبيعى الخامس NF٥ كالآتى:

أولاً : تحديد فئة التفكيكات المحتملة لكى تأخذ فى الاعتبار علاقة الربط R. ولتحديد هذه الفئة ، ينبغى تعريف كل التفكيكات الممكنة.

ثانيًا : يتم تحليل كل تفكيك محتمل بالسؤال التالى:

"(أ) متى يوجد واقعة لكل علاقة ربط مفككة؟

(ب) وأى وظيفة مشتركة لعلاقتى ربط أو أكثر تتضمن نفس الكينونة فى كل واقعة ربط؟ وعندئذ هل يتحقق ارتباط علاقة الربط الأصلى n-ary بين وقائع الكينونة المشاركة؟". فى حالة الإجابة بـ "نعم". عندئذ يمكن تفكيك علاقة الربط R إلى ثلاثة علاقات ربط أو أكثر يتم تعريفها بواسطة هذا التفكيك المحتمل.

(٤) فى حالة (اتمام تعريف التفكيك فى الخطوات من ١ إلى ٣ ، فإنه ينبغى إكمال تعريف علاقات الربط المفككة كما فى الخطوات من ٥ إلى ٨. بخلاف ذلك لا يمكن التفكيك.

(٥) يتم اختيار أسماء علاقات الربط المفككة المبنية على الدلاليات.

(٦) يتم إضافة أى قيود تكون محمية بواسطة العلاقات المفككة للنموذج.

(٧) يتم تخصيص عدد من الخصائص (من صفر فأكثر) المعرفة لعلاقة الربط R لواحدة من علاقات الربط المفككة أو لواحدة من أنواع الكينونات المشاركة. ويتم تخصيص كل خاصية لنوع الكينونة أو لعلاقة الربط التى لها مفتاح وظيفى يحدد الخاصية. بمعنى آخر ينبغى أن يوجد قيمة واحدة فقط للخاصية وذلك لأى واقعة محددة لنوع الكينونة أو لعلاقة الربط. ولو كانت الخاصية التى تم تخصيصها لعلاقة الربط الثنائية تتضمن القيد Cmax (a,b) = 1 فإنه يتم تخصيص الخاصية للوظيفة a بدلاً من نوع الكينونة.

(٨) يجب مراجعة نموذج البيانات كله لتحديد ما إذا كانت علاقة الربط مكافئة لأى من العلاقات المفككة الموجودة من قبل. وتكون علاقتيا الربط متكافئتين فى حالة تضمينهما نفس أنواع الكينونات المشاركة بنفس المعنى الدلالى، وإن كل نوع كينونة مشاركة ينبغى تضمينه فى نفس عدد الوظائف فى كلا علاقتى الربط. ويحدث تكافؤ لعلاقات الربط ، إذا كانت علاقة الربط الاصلية R، ونفس علاقة الربط الزائدة عن الحد جزئيًا (قبل تفكيك علاقة الربط R) قد تم أخذها فى الاعتبار. وبالتالى فى حالة اكتشاف تكافؤ بين علاقتى الربط ، يتم تخصيص كل الخصائص لإحدى علاقات الربط ، وتحذف الأخرى من نموذج البيانات.

تطبيقات توضيحية :

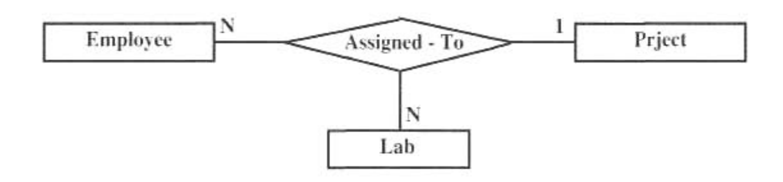
• تفكيك علاقة الربط "يقدم فاتورة" Bills من الشكل رقم (٦-٢٤) إلى الشكل رقم (٦-٢٦) مبنى على جدول التعددية فى الشكل رقم (٦-٢٩) وهو تطبيق مباشر للخطوة (١). ويوجد احتمال تفكيك واحد فقط بسبب القيد Cmax (I , C). وعلاقة الربط R1 تتضمن

العميل Customer والفاتورة Invoice. وعلاقة الربط R2 تتضمن الفاتورة Invoice والمنتج Product. والنموذج الاصلى يتضمن فقط قيدين يجب أن يكونا محميين هما:

Cmax (I,C) = 1 , Cmax (IP,C) =1

والنموذج فى شكل رقم (٦-٢٦) تم استنتاجه باستعمال قاعدة الاكتمال؛ لذلك فكل القيود تكون محمية.

شكل رقم (٦-٣٢) تخصيص الموظفين Employees لمعامل Labs المشاريع Project



• علاقة الربط "تخصص" فى الشكل رقم (٦-٣٢) تعرض مثالاً توضيحياً اكثر تعقيداً. هذه العلاقة تبين تخصص الموظفين Employees للمعامل Labs فى المشاريع الحالية Projects. كل موظف "يعمل فى" Works-On فى مشروع واحد فقط فى المرة. كل معمل Lab يخصص لمشروع واحد فقط فى المرة. كل مشروع يمكن أن ينتشر عبر ثلاثة معامل كحد أقصى فى المرة. بمجرد إتمام التخصيصات لكل موظف ، يتم حذفها من قاعدة البيانات وتحل بتخصيصات جديدة. ويعكس جدول التعددية فى شكل رقم (٦-٣٣) قواعد العمل. يتم استخلاص القيود التالية من جدول التعددية فى الشكل رقم (٦-٣٣) وتخصيصها إلى وظائف علاقة الربط الملائمة فى شكل رقم (٦-٣٤أ) المبنى على تقليديات نموذج كينونة - علاقة ER الطبيعية:

(a1) Cmax (E,P) = 1

(a2) Cmax (E,L) = 3

ويتم أيضاً استخلاص تلك القيود من جدول التعددية:

Cmax (P,E) = N

Cmax (L,E) = N

شكل رقم (٦-٣٣) جدول التعددية لعلاقة الربط "يخصص"

	S	P	L
Employee		1	3
Project	N		3
Lab	N	1	

EP	EL	PL
	N	3
N	N	N
N	N	

EP		N	3
EL	N	1	N
PL	1	N	

التى يتم استعمالها لعلاقات الربط المفككة ، ولكن ليس معنوياً لتحليل حماية القيد. بالإضافة إلى القيود المحمية بواسطة التفكيك المحتمل التى يتم استنتاجها كالآتى:

باستعمال قاعدة الاكتمال من a1 يكون الناتج

(a3) Cmax EL,P) = 1

باستعمال قاعدة الاكتمال من a2 يكون الناتج :

(a4) Cmax (EP,L) = 3

باستعمال قاعدة الانتقال الكاذبة للقيد a1 ، a4 يكون الناتج :

(a5) Cmac (E,PL) = 3

والقيود التالية ليست محمية بواسطة التفكيك المحتمل :

Cmax (P,L = 3

Cmax (L,P) = 1

يؤدى القيد Cmax (L,P) = 1 فى الشكل رقم (٦-٣٣) إلى التفكيك المحتمل فى الشكل رقم (٦-٣٤ب) التى تتضمن القيود التالية:

(b1) Cmax (L,P) = 1

(b3) Cmax (P,L) = 3

(b3) Cmax (E,L) = 3

بالإضافة إلى القيود المستنتجة التالية:

باستعمال قاعدة الاكتمال من b1 ينتج

(b4) Cmax (EL,P) = 1

باستعمال قاعدة الاكتمال من b2 ينتج

(b5) Cmax (EP,L) = 3

باستعمال قاعدة الانتقالية من b1 ، b3 ينتج :

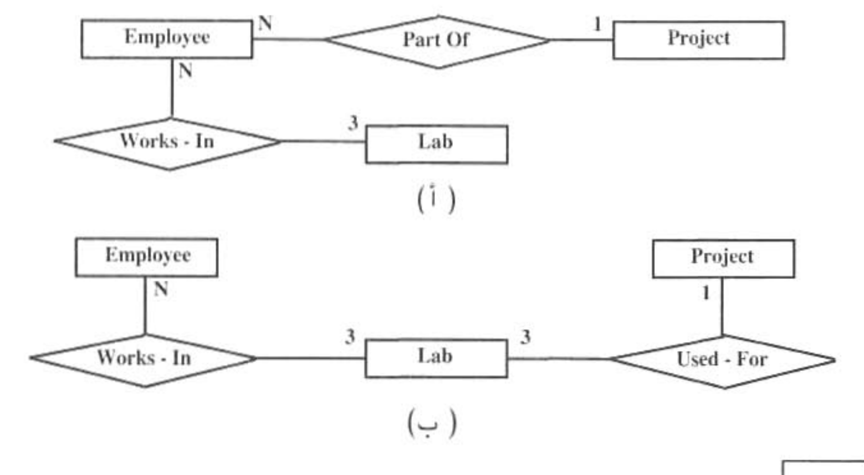
(b6) Cmax (E,PL) = 3

القيود التالية ليست محمية بواسطة التفكيك المحتمل :

Cmax (E,P) =1

تعرف الخطوة (١) التفكيك فى هذا المثال ، أما الخطوات من (٥) إلى (٨) يتم استعمالها لتكمل علاقات الربط المفككة. وبناء على الخطوة (٦) فإن القيد Cmax (E,P) = 1 ينبغى أن يكون مضافاً إلى النموذج المفكك حيث توجد القيود محمية.

شكل رقم (٦-٣٤) يوضح التفكيكات المحتملة لعلاقة الربط "يخصص"



نموذج كينونة - علاقة المطور Enhanced Entity- Relationship (EER) :

ظهرت العديد من التوسعات لنموذج كينونة – علاقة. بدء العمل فى هذه التوسعات عرف بالصفات الثانية لنموذج كينونة علاقة ربط ER. ومعظم المفاهيم المهمة فى نمذجة البيانات الدلالية يمكن أن يتم تمثيلها بشكل مناسب على نموذج كينونة – علاقة المطور EER. وتتضمن الإضافات المهمة لذلك النموذج الأنوا ع الأصلية Superclass والأنوا ع الفرعية Subclass والآلية المرتبطة بإحكام والمعروفة بوراثة الخاصية attribute in-heritance. ويظل نموذج كينونة – علاقة المطور EER نموذجًا رئيسيًا لتصميم المخطط المفاهيمى. وكان أحد الأسباب المهمة للانتشار الواسع لهذا النموذج هو توفيره لتقنية أعلى – أسفل لتصميم قاعدة البيانات، والتى تؤدى إلى توظيف مفهوم التجريد Abstraction. ويلاحظ أنه فى حالة تجاهل التمييز بين أنوا ع الكينونات وأنوا ع علاقات الربط فى النموذج فإنه يتحول إلى نموذج عام لأنوا ع الأشياء Object classes. وسوف يتم سرد مفاهيم نموذج كينونة – علاقة المطور بشىء من التفاصيل[١]،[٦]،[٢].

مفاهيم نموذج كينونة - علاقة المطور EER :

* الأنوا ع الأصلية والأنوا ع الفرعية Superclasses & Subclasses :

قد تم توضيح أن نوع الكينونة يتم استعماله لتمثيل كينونات لها نفس النوع مثل مجموعة كينونات الموظف فى مثال (٦–٥) لقاعدة بيانات شركة المشاريع الهندسية (الافتراضية). ولكن فى حالات كثيرة يكون نوع الكينونة له مجموعات فرعية عديدة لكينوناته ذات معنى، وتحتاج إلى تمثيلها بشكل واضح. على سبيل المثال: الكينونات الأعضاء فى نوع الكينونة الموظف قد تكون مجمعة فى أكثر من مجموعة مثل الفنيين Technician والمديرين Manager والسكرتارية Secretary والمهندسين Engineer وغيرها. وكل من هذه المجموعات تنتمى إلى نوع الكينونة "الموظف" EMPLOYEE وهو ما يعرف بالمجموعات الفرعية أو النوع الفرعى Subclass لنوع الكينونة "الموظف" EMPLOYEE وهو ما يعرف بالنوع الأصلى Superclass لكل من هذه الأنوا ع الفرعية. وتسمى علاقة الربط بين النوع الأصلى وأى نوع فرعى له باسم علاقة ربط نوع أصلى/فرعى Super-class / Subclass relationship. وعندما يتم تطبيق علاقة الربط نوع أصلى/فرعى فى

قواعد البيانات، فإنه قد يتم تمثيل أعضاء النوع الفرعي كشيء Object مميز أو سجل record مميز يرتبط بكينونة النوع الأصلي عبر خاصية المفتاح(٧)،(٨) .

وهناك مفهوم مهم مرتبط بالنوع الفرعي هو وراثة الخاصية: لأن كل كينونة في النوع الفرعي يتم تمثلها بكينونة في العالم الحقيقي من النوع الأصلي. وإنه ينبغي معالجة قيم خصائصها كنوع الفرعي، بالإضافة إلى قيم خصائصها كعضو (ككينونة) في النوع الأصلي. والكينونة التي تكون عضواً في النوع الفرعي ترث كل خصائص الكينونة العضو في النوع الأصلي، وترث أيضاً كل وقائع علاقة الربط لأنواع علاقة الربط التي يشارك فيها النوع الأصلي.

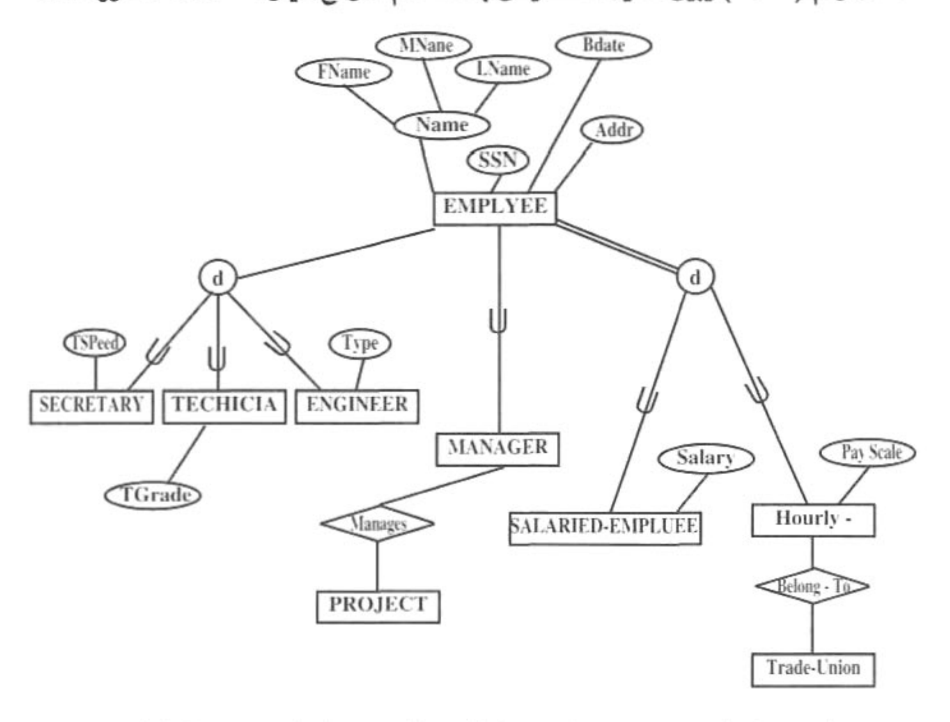
* **التخصيص Specialization :**

التخصيص هو عملية تعريف مجموعة الأنواع الفرعية Subclasses لنوع كينونة معين ، ونوع الكينونة هذا يسمى النوع الأصلي للتخصيص. فعلى سبيل المثال: مجموعة الأنواع الفرعية (السكرتارية و المهندسون و الفنيون) هو تخصيص لنوع الكينونة الأصلي "الموظف" التي تتميز بين كينوناته بناء على نوع وظيفة كل عضو. وقد يوجد العديد من التخصيصات التي يتم بناؤها لنفس نوع الكينونة بناء على صفات تمييزية مختلفة. على سبيل المثال: تخصيص آخر لنوع الكينونة "الموظف" قد يحقق مجموعة أنواع فرعية (مرتبات الموظفين Salaried-Employee و ساعات عمل الموظفين Hourly-Employee) ٢٦١ وهذا التخصيص يميز الموظفين بناء على طريقة الدفع.

ويبين الشكل رقم (٦-٣٥) تمثيل التخصيص برسم تخطيطي في نموذج كينونة- علاقة المطور EER. والأنواع الفرعية التي تعرف تخصيص معين يتم إلحاقها بواسطة دائرة تتصل بالنوع الأصلي يوضع رمز الفئة الجزئية (⊂) على كل خط يصل النوع الفرعي بالدائرة حيث يتم الإشارة إلى اتجاه علاقة الربط النوع الأصلي/الفرعي. والخصائص التي توظف فقط لكينونات نوع فرعي معين ، تلحق بالمستطيل الذي يمثل النوع الفرعي وتسمى "خصائص محددة" للنوع الفرعي. والنوع الفرعي يمكن أن يشارك في أنواع علاقات ربط محددة مثل نوع ساعات عمل الموظفين Hourly-Employee التي تشارك في علاقة الربط "ينتمي إلى" Belong-To كما في الشكل رقم

(٦-٣٥). وكما يبين الشكل رقم (٦-٣٦) وقائع الكينونة التى تنتمى إلى الأنواع الفرعية (السكرتارية Secretary والمهندسين Engineer والفنيين Technician) للتخصيص. وعلاقة الربط نوع أصلى/فرعى مثل موظف/سكرتارية Employee / Secretary التى تشبه إلى حد بعيد علاقة الربط (١:١) لمستوى الواقعة [١]،[٢].

شكل رقم (٦-٣٥) يبين تمثيل لتخصيص باستخدام نموذج كينونة - علاقة المطور EER



وهناك سببان لاستخدام علاقات ربط النوع الفرعى داخل نموذج البيانات :

(١) بعض الخصائص قد تلحق فقط ببعض الكينونات للنوع الأصلى وليس لكل الكينونات .

(٢) بعض أنواع علاقات الربط قد يتم مشاركته فقط بواسطة كينونات تكون أعضاء للنوع الفرعى .

* **التعميـم Generalization :**

ومما سبق يتضح أن عملية التخصيص تسمح بالآتى:

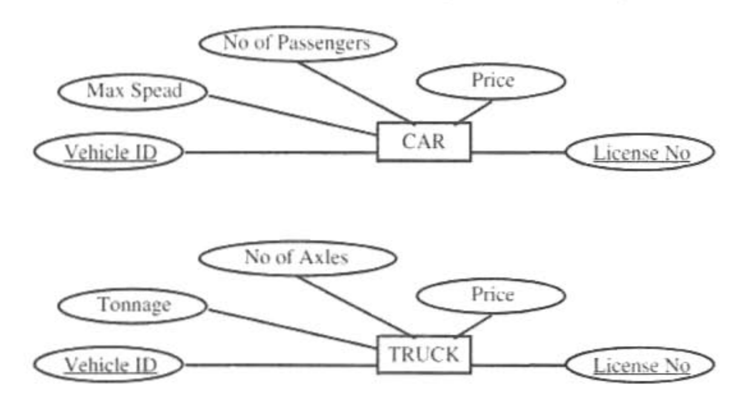
١- تعرف مجموعة أنواع فرعية لأى نوع كينونة.

٢- ارتباط خصائص محددة إضافية بأنواع علاقات الربط لكل نوع فرعى.

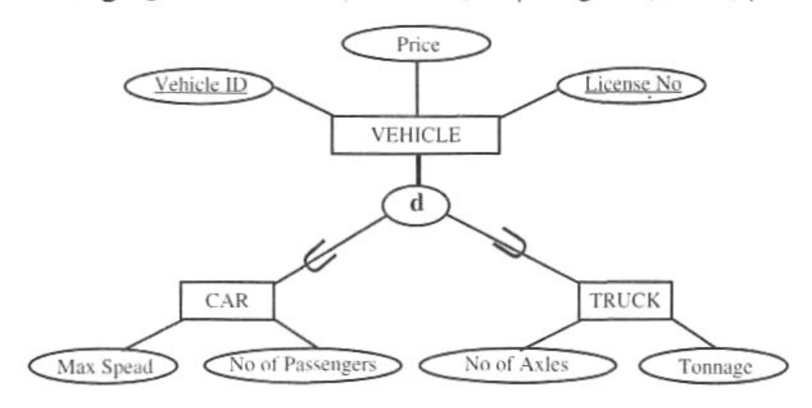
٣- إنشاء أنواع علاقات ربط محددة إضافية بين كل نوع فرعى وأنواع الكينونات الأخرى.

تعتبر عملية **التعميـم** عكس عملية التجريد. وفيها يتم وضع حدٍ للاختلافات بين أنواع الكينونات العديدة ، ويؤدى إلى تعريف معالمها الشائعة ويعممها فى نوع أصلى واحد لنوع الكينونة التى لها أنواع كينونات أصلية هى بمثابة أنواع فرعية خاصة. فعلى سبيل المثال: يعتبر أنواع كينونات "السيارة" CAR و"الشاحنة" TRUCK كما فى الشكل رقم (٦-٣٦أ) والتى قد تم تعميمهما فى نوع كينونة "المركبة" VECHILE فى الشكل رقم (٦-٣٦ب). وكلاهما نوعان فرعيان لنوع أصلى تم تعميمه هو "المركبة" VE-CHILE. ويستخدم مصطلح "تعميم" للإشارة إلى عملية تعريف نوع كينونة تم تعميمه من أنواع الكينونات المتواجدة[١]،[٢].

شكل رقم (٦-٣٦أ) يوضح نوعى كينونة السيارة والشاحنة



شكل رقم (٦-٣٦ب) يوضح تعميم نوعى كينونة "السيارة" و "الشاحنة" فى نوع كينونة "المركبة"



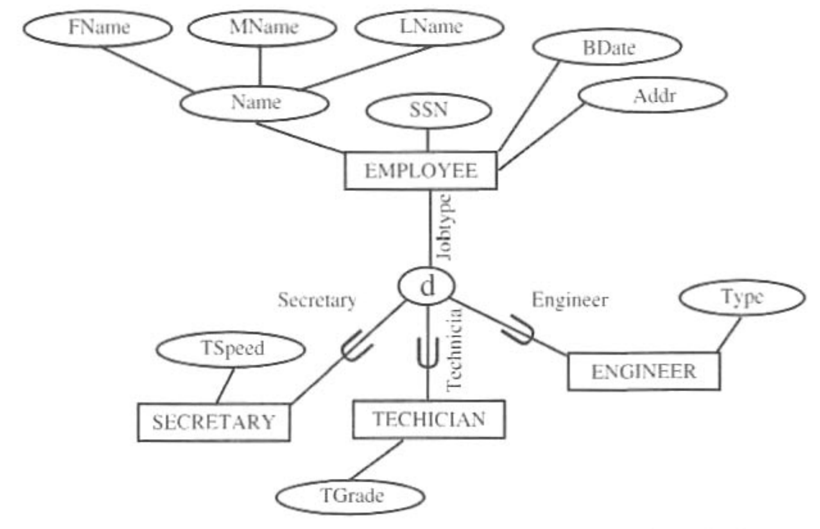
وينظر إلى عملية التعميم على أنها عكس عملية التخصيص ، ولذا فان الشكل رقم (٦-٢٣ب) يبين الفئة المكونة من {لسيارة CAR ، الشاحنة} TRUCK كتخصيص لنوع كينونة "المركبة" VECHILE اكثر منها تعميم لنوعى كينونة "السيارة" CAR و"الشاحنة" TRUCK. وأيضاً يمكن النظر إلى نوع كينونة "الموظف" EMPLOYEE كتعميم لأنوا ع الكينونات الفرعية السكرتارية Secretary والمهندسين Engineer والفنيين .Technician يتم استعمال رمز الفئة الجزئية (⊂) كسهم يستخدم للإشارة إلى النوع الأصلى الذى تم تعميمه فإنه يمثل تعميماً. عندما يتم استخدامه كسهم للإشارة إلى النوع الفرعى الذى تم تخصيصه فانه يمثل تخصيصاً.

قيود على التخصيص والتعميم Constraints on specialization and Generalization :

فى بعض التخصيصات يمكن تحديد الكينونات التى ستصبح أعضاء لكل نوع فرعى، وذلك بوضع شرط على قيمة خاصية معينة للنوع الأصلى. مثل هذه الأنوا ع الفرعية تسمى "شرط التعريف" Condition-defined للأنوا ع الفرعية. فعلى سبيل المثال: لو أن نوع كينونة "الموظف" له خاصية نوع الوظيفة JobType كما هو موضح فى الشكل رقم (٦-٣٧). فإنه يمكن تحديد العضوية فى النوع الفرعى "سكرتارية" SECRETARY بوضع الشرط الخاص بالتساوى (Jobtype = "Secretory")، وهذا الشرط هو القيد الذى يحدد عضوية النوع الفرعى "سكرتارية" SECRETARY، والذى يجب أن يحقق الشرط، وإن كل الكينونات فى نوع الكينونة "الموظف" EMPLOYEE التى لها قيمة الخاصية

"سكرتارية" الخاصة بخاصية نوع الوظيفة JobType يجب أن تنتمى إلى النوع الفرعى "سكرتارية" SECRETARY.

الشكل رقم (٦ – ٣٧) تخصيص ذات "شرط التعريف" على خاصية الموظف "نوع الوظيفة"



ولو أن كل الأنواع الفرعية فى التخصيص لها شرط العضوية على نفس الخاصية فى النوع الأصلى، فإن التخصيص ذاته يسمى تخصيص "معرف الخاصية" Attribute-defined والخاصية تسمى خاصية التعريف.

وفى حالة عدم وجود مثل هذا الشرط الذى يحدد العضوية ، فإن النوع الفرعى يسمى "تعريف – المستفيد" User defined. وتحدد العضوية فى مثل هذا النوع الفرعى بواسطة مستخدمى قاعدة البيانات عند تطبيق عمليات إضافة كينونة إلى النوع الفرعى. وهناك قيدان آخران يمكن تطبيقهما على التخصيص، هما:

١– قيد التفكيك Disjointness Constraint :

يحدد هذا القيد النوع الفرعى للتخصيص الذى يجب أن يتم تفكيكه disjoint . بمعنى أن الكينونة يمكن أن تكون عضواً على الأكثر لواحد من الأنواع الفرعية للتخصيص. وإذا كانت الأنواع الفرعية غير مفككة ، فإن مجموعات كينوناتها ربما

تتداخل Overlap ، بمعنى آخر فإن الكينونة نفسها قد تكون عضواً لأكثر من نوع فرعى فى التخصيص. ويمكن استخدام الرمز (d) داخل دائرة صغيرة لتحديد القيد "تعريف - المستفيد" للأنواع الفرعية للتخصيص التى يجب أن تكون مفككة. وكذلك الرمز (O) داخل دائرة صغيرة للأنواع الفرعية للتخصيص التى قد تكون متداخلة.

٢. قيد حد الكمال Completeness Constraint:

يحدد التخصيص الكلى total القيد الذى يجب أن تكون كل كينونة فى النوع الأصلى عضواً لنوع فرعى معين فى التخصيص. فعلى سبيل المثال: فى حالة وجوب ان يكون كل موظف إما فى مرتبات الموظفين Salaried-Enployee أو عدد ساعات الموظفين Hourly-Employee ، فإن ذلك التخصيص يكون تخصيصاً كلياً للموظف. ويعبر عنه فى الرسم التخطيطى لنموذج كينونة - علاقة ربط المطور EER باستخدام الخط المزدوج .

ويسمح التخصيص الجزئى للكينونة ألا تنتمى الكينونة إلى أى أنواع فرعية. فعلى سبيل المثال: لو أن كينونة موظف Employee لم تنتمِ إلى أى من الأنواع الفرعية عندئذ يكون التخصيص جزئياً: ومما سبق يتضح أن التخصيص يمكن تصنيفه إلى:

- تخصيص مفكك (كلى- جزئى).
- تخصيص متداخل (كلى - جزئى).

أما تعميم النوع الأصلى عادة يكون كلياً: لأن النوع الأصلى يستنتج منه الأنواع الفرعية.

قواعد الإضافة والحذف:

تطبق قواعد الإضافة والحذف للتخصيص والتعميم؛ بناء على قيود محددة مسبقاً، وبعض هذه القواعد هى:

١- حذف أى كينونة من النوع الأصلى يتضمن حذفها تلقائياً من كل الأنواع الفرعية التى تنتمى إليها.

٢- إضافة أى كينونة إلى النوع الأصلى تتضمن إضافتها إجباريًا إلى كل الأنوا ع الفرعية التى تحقق "شرط التعريف".

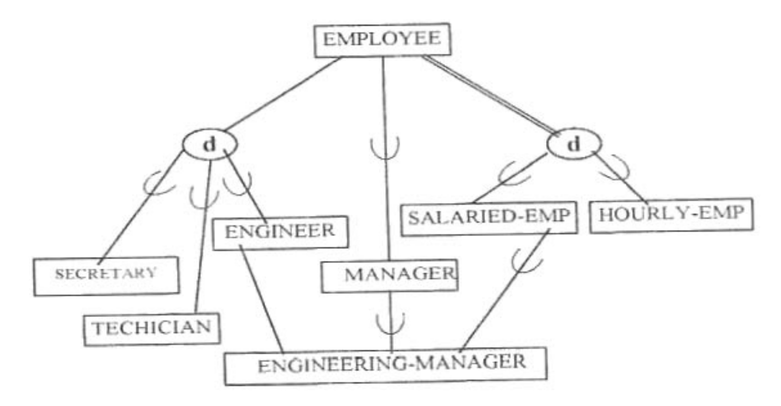
٣- إضافة أى كينونة فى النوع الأصلى لأى تخصيص كلى تتضمن إضافة الكينونة إجباريًا على الأقل فى واحدة من الأنوا ع الفرعية للتخصيص.

هرمية وتشابكية التخصيص:

قد يتفرع من النوع الفرعى أنوا ع فرعية ، تشكل تخصيصات هرمية أو تشابكية كما هو مبين فى الشكل رقم (٦ – ٣٨) فعلى سبيل المثال: نوع كينونة "المهندس" ENGINEER هو نوع فرعى لنوع كينونة "الموظف" EMPLOYEE وهو أيضًا نوع أصلى لنوع كينونة "مدير قسم الهندسة" .Engineering-Manager وهذا يمثل قيد العالم الحقيقى ، حيث إن كل مدير قسم هندسى قد يطلب منه أن يعمل مهندسًا. وهرمية التخصيص لديها قيد على أن كل نوع فرعى يشارك فى علاقة ربط واحدة لنوع أصلى/فرعى فى حين أنه فى تشابكية التخصيص فإن النوع الفرعى يمكن أن يكون نوعًا فرعيًا فى أكثر من علاقة ربط لنوع أصلى/فرعى. أما النوع الفرعى المشارك هو نوع فرعى يشارك مع أكثر من نوع أصلى. فعلى سبيل المثال:

فى حالة أن كل مدير قسم هندسة قام بالعمل مهندسًا ، يجب أيضًا أن يتقاضى مرتبًا كموظف Salaried-Employee. عندئذ ينبغى أن يشارك مدير قسم الهندسة فى نوع فرعى لكل من الأنوا ع الأصلية الثلاثة فى الشكل رقم (٦-٣٨) ، ويلاحظ أن الأنوا ع الفرعية المشاركة تؤدى إلى التشابكية ، أى أنه لو لم توجد أنوا ع مشاركة فإن الهرمية تصبح أكثر من التشابكية. ويتشابه أيضًا تطبيق مفاهيم التعميم من حيث هرمية وتشابكية التعميم كما فى التخصيص[1].

شكل رقم (٦ - ٣٨) تشابكية التخصيص مع النوع الفرعى المشارك "مدير قسم الهندسة"



أحد الأساليب الاساسية لحماية الدلاليات، العناية بهياكل الهرمية والتى تحتوى على علاقة الربط "يكون" Is-A أثناء عنقودية Clustring. وفى بعض الحالات يتم الاحتفاظ بكل الهرم فى نفس العنقود. وفى بعض الحالات الهرم يتم تقسيم الهرم إلى عناقيد مختلفة. وعلى المصمم اتخاذ القرار الملائم على أساس الدلاليات المحددة لعناصر المخطط[١١].

التصميم المفاهيمى أعلى - أسفل ، وأسفل - أعلى:

التخصيص المتتالى يناظر عملية التكرير المفاهيمى أعلى - أسفل أثناء تصميم المخطط المفاهيمى فى حين أن عملية التعميم تناظر التحليل المفاهيمى أسفل - أعلى.

المصنفات والتصنيف Categories & Categorization :

كل علاقات الربط نوع أصلى / فرعى لها على الأكثر نوع أصلى واحد حتى النوع الفرعى المشارك مثل "مدير قسم الهندسة" Engineering-Manager فى التشابكية فى الشكل رقم (٦ - ٣٨) هو نوع فرعى فى ثلاثة علاقات ربط نوع فرعى / أصلى مميزة؛ وذلك لأن كلاً من علاقات الربط الثلاثة لها نوع أصلى واحد. فى بعض الحالات تظهر حاجة ملحه لنمذجة علاقة ربط نوع أصلى/ فرعى مع أكثر من نوع أصلى (حيث إن تمثيل الأنواع الأصلية يختلف عن أنواع الكينونات) ، وفى مثل هذه الحالة يسمى النوع الفرعى مصنفًا Category[١] .

مثال (٦-٦):

يفترض أن هناك ثلاثة أنواع كينونات فى قاعدة بيانات خاصة بتسجيل المركبات، هى: "الشركة" COMPANY و "البنك" BANK و"الشخص" PERSON. وصاحب أى مركبة قد يكون شخصاً Person أو بنكًا Bank أو شركة Company.

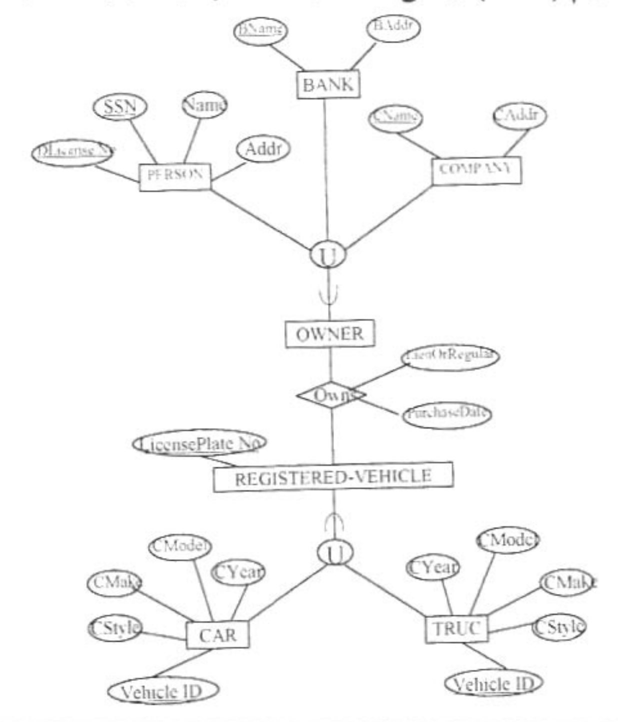
المطلوب :

إنشاء نوع Class يتضمن أنواع الكينونات الثلاث لتمثل وظيفة (دور) صاحب المركبة.

الحـل :

يتم إنشاء المصنف "صاحب" OWNER الذى يكون نوعًا فرعيًا لاتحاد Union الأنواع الثلاثة. وتظهر المصنفات فى نموذج كينونة - علاقة المطور EER كما فى الشكل (٦-٣٩) حيث إن الأنواع الأصلية الثلاثة (بنك، شخص، شركة) متصلة بدائرة صغيرة بداخلها رمز الاتحاد (U)، الذى يشير إلى عملية اتحاد الفئات.

شكل رقم (٦-٣٩) يوضح المصنفين "صاحب" و "المركبة المسجلة"



وأى مصنف له نوعان أصليان أو أكثر قد يتم تمثيلهما بأنواع كينونات مميزة، فى حين أن علاقات ربط الأنواع الأصلية/الفرعية دائماً لها نوع أصلى واحد.

ويمكن عمل مقارنة بين المصنف والنوع الفرعى المشارك. و يمكن صنع هذه المقارنة بين المصنف "صاحب" OWNER والنوع الفرعى المشارك "المركبة المسجلة" Engineering - Manager.

ENGINEERING - MANAGER	OWNER
* هو نوع فرعى لثلاثة أنواع أصلية هى: "المدير" MANAGER و "المهندس" EN-GINEER و "مرتبات الموظفين" Salaried-Employee.	* هو نوع يتضمن ثلاثة أنواع من الكينونات، هى: "البنك" BANK و "الشركة" COMPANY و "الشخص" PERSON .
* هذا النوع هو فئة جزئية لتقاطع Inter-section ثلاثة أنواع فرعية.	* هذا المصنف هو فئة جزئية لاتحاد Union أنواع أصلية .
* يورث هذا النوع كل خصائصه لأنواعه الأصلية التى يشاركها.	* كل كينونة لهذا النوع سوف تورث الخصائص لنوع واحد فقط من أنواع الكينونات التى تتضمنها، معتمداً على النوع الأصلى التى تنتمى إليه الكينونة.

يمكن أن يكون المصنف كليًا أو جزئيًا. ويلاحظ أنه لو كان المصنف كليًا فإنه قد يتم تمثيله كتخصيص أو تعميم. أما فى حالة وجود نوعان يمثلان نفس نوع الكينونات ويشاركان صفات عديدة تتضمن نفس المفتاح ، عندئذ يفضل عمل التعميم ، بخلاف ذلك يستحسن عمل التصنيف.

مخطط قاعدة البيانات فى نموذج كينونة – علاقة المطور EER :

يمكن توضيح مختلف المفاهيم السابقة لمخطط قاعدة بيانات فى نموذج كينونة – علاقة المطور EER من خلال المثال رقم (٦-٧).

مثال (٦-٧):

يفترض أن تحتفظ قاعدة بيانات جامعة معينة بالبيانات التالية كما هو موضح بالشكل رقم (٦-٤٠):

الطلاب Students، تخصصاتهم Majors، درجاتهم Transcripts التسجيل Registrations بالإضافة إلى فرص دراسة المناهج التعليمية Course offering.

* تحتفظ قاعدة البيانات أيضًا بمشروعات البحث research projects وبيانات الطلاب الخرجين Graduate Students .

* خصائص نوع الكينونة PERSON هي:

الاسم Name ، رقم التأمين الاجتماعي SSN ، العنوان Address ، الجنس Sex ، تاريخ الميلاد BDate.

■ نوعان فرعيان لنوع الكينونة PERSON هما:

- الطالب STUDENT

- الكلية FACULTY .

* خصائص النوع الفرعي "كلية" FACULTY هي:

* المرتبة العلمية RANK وتحتوي على: مساعد assistant ، مرتبط associate ، ملحق adjunct ، بحث research ، زيارة Visiting.

* مكاتب الكلية Foffice .

* تليفونات الكلية Fphone .

* المرتبات Salary .

* أعضاء الكلية المرتبطين بأقسام هيئة التدريس قد يرتبطوا بأكثر من قسم؛ ولذا فإن علاقة الربط هي "ينتمي" Belong وهي علاقة ربط متعدد - لمتعدد .(N : M)

* خصائص النوع الفرعى "طالب "STUDENT هى:

– السنة الدراسية Class وتحتوى على:

√ طالب جديد Freshman

√ طالب بالسنة الثانية Sophomore

√ طالب بالسنة الثالثة Junior

√ طالب بالسنة الرابعة Senior

√ طالب خريج Graduate Student.

♦ يلتحق الطالب بقسمه التخصيصى Major أو بقسم إضافى minor .

♦ يختار الطالب أجزاء من المناهج التى يريد أن يدرسها ويسجلها.

♦ وتحدد كشوف درجات الطلاب المكتملة في نوع الكينونة "درجات الطلاب" TRANSCRIPTS وتوجد كل درجة فى نوع كينونة "التقدير" GRADE الذى يتسلمه الطالب فى أجزاء المناهج.

* يعتبر النوع "الدارس" Grad-Student نوعاً فرعياً "للطالب" STUDENT مع تعريف الشرط Class = 5.

* تحتفظ قائمة الدرجات السابقة لكل خريج فى خاصية متعددة القيم و مركبة هى "الدرجات" Degrees .

* يرتبط الطالب بمشرف الكلية Advisor واللجنة Committee (إن وجدت).

* قسم أعضاء هيئة التدريس له الخصائص التالية: الاسم DName، التليفون DPhone ورقم المكتب Office وهو يرتبط برئيس القسم Chiairs.

* وكل كلية لها مجموعة خصائص، منها: اسم الكلية CName ، رقم المكتب COffice ، مكتب العميد Dean.

* المادة Course لها مجموعة خصائص ، منها: رقم المادة #C، اسم المادة CName وصف المادة CDesc.

* والعديد من القاعات Sections تخصص لكل مادة، وكل قاعة لها مجموعة خصائص، منها: رقم القاعة # Sec، واسم المادة CName، السنة الدراسية year الفصل الدراسى Qtr ورقم القاعة لا يتكرر.

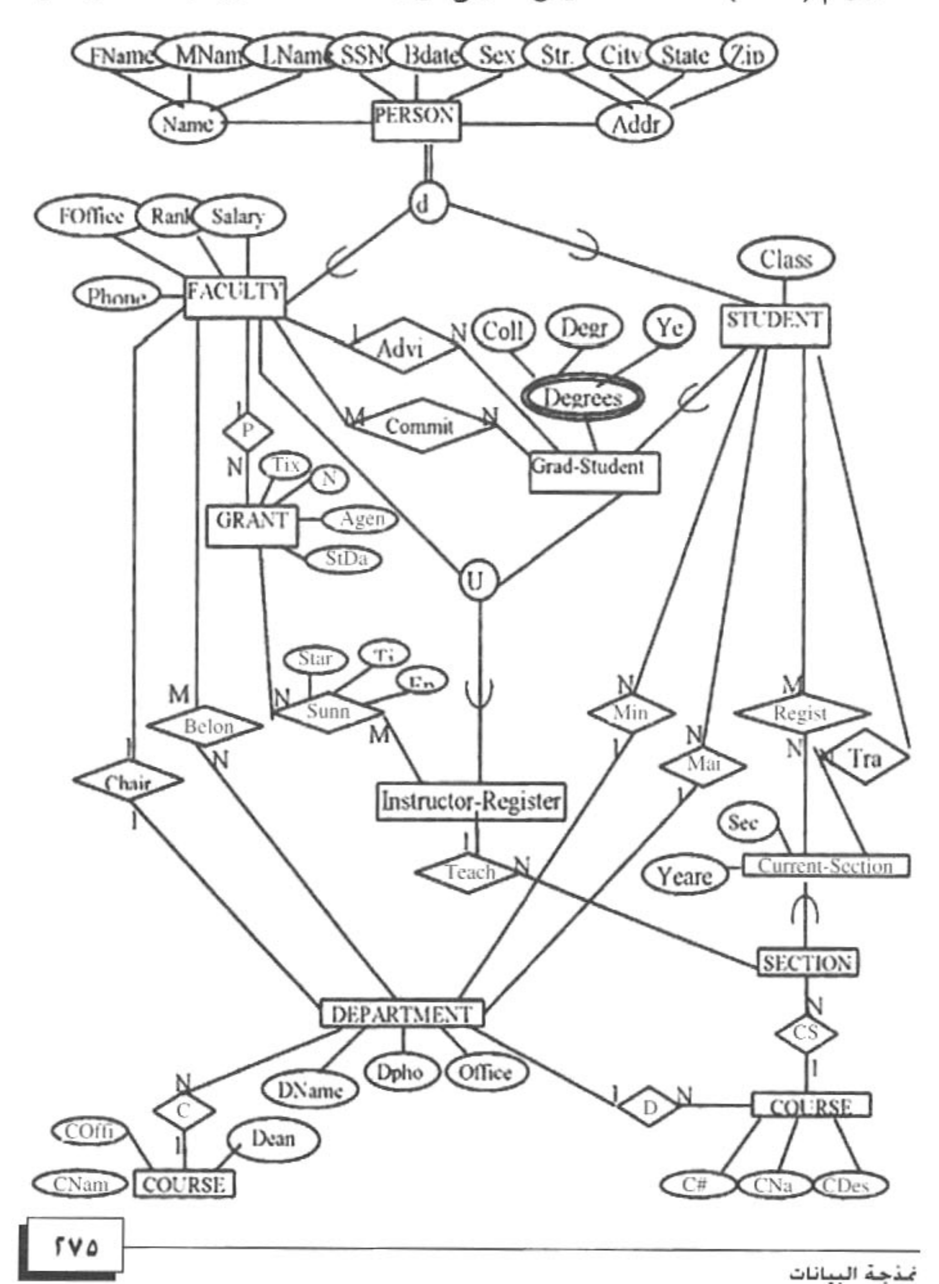
* يتضمن النوع الفرعى "القاعة الحالية" Current-Section القاعات Sections بشرطى التعريف Qtr = Current Qtr ، Year = Current year . وكل قاعة ترتبط بالمحاضر Instructor الذي يدرس فيها.

* والمصنف " محاضر- باحث" Instructor-Researcher هو فئة جزئية لاتحاد نوعى الكينونة "الكلية" Faculty و "الدارس" Grad-Student ويتضمن كل الكليات بالإضافة إلى الخرجين الذين تم تدعيهم بتدريس أو بحث.

* أما نوع الكينونة "المنح" GRANT تحتفظ بالمنح البحثية والعقود الممنوحة للجامعة ولها الخصائص التالية: عنوان المنحة Title، رقمها No ، الجهة المانحة Agency وتاريخ البداية StDate.

* والمنحة ترتبط بأحد الباحثين الرئيسيين (P1) وكل الباحثين الداعمين Support. وكل واقعة للدعم Support لها خصائص معينة ، هى : تاريخ بداية الدعم Start ونهاية تاريخ الدعم end ، الوقت المنقضى حاليًا فى المشروع Time بواسطة الشخص المدعم.

شكل رقم (٦-٤٠) المخطط المفاهيمى لنموذج كينونة - علاقة المطور لجامعة افتراضية



المبادئ الأساسية لنمذجة البيانات الدلالية لتطبيقات قواعد البيانات:

توجد مجموعة من المفاهيم التجريدية التى تستخدم فى نماذج البيانات الدلالية منها[١]،[٩]:

* تصنيف/التفريع Classification / Instantiation.
* التعريف Identification.
* التعميم/التخصيص Generalization / Specialization.
* التجميع aggregation.
* الارتباط association.

المفاهيم المزدوجة كل عكس الأخر مثل مفهوم تصنيف/تفريع ، ومفهوم تعميم/ تخصيص. بالإضافة إلى المفاهيم المترابطة مثل التجميع والارتباط. وينبغى أن تدعم بكل هذه المفاهيم التجريدات الرئيسية لنموذج البيانات الدلالية. ويتم تطبيق هذه التجريدات البيانية بشكل متساوٍ مع نماذج البيانات الشيئية الموجة.

التجميع والارتباط Aggregation & Association :

التجميع هو مفهوم تجريدى لبناء تجميع الأشياء من محتوى الأشياء. وعلاقة الربط بين الترتيب الأقل للشىء والترتيب - الأعلى له يتم توصيفها كعلاقة ربط "يكون جزءًا من" Is-Part - Of. وفى مستوى آخر يستعمل هذا التجريد لتجميع الخصائص فى الشىء. ونموذج كينونة - علاقة ER التجميعى يتم استعماله عندما يوجد اثنان أو أكثر من أنواع الكينونات يمكن ربطهما لتعريف نوع علاقة ربط.

مفهوم تجريد الارتباط يستعمل لربط الأشياء من أنواع عديدة مستقلة تشابه فى استعمال التجميع فى هذه الحالة فقط. ويتم تمثيلها فى نموذج كينونة – علاقة المطور EER بواسطة أنواع علاقات الربط.

– التعريف Identification :

إنه يشير إلى العملية التى بواسطتها يتم تجرد المفاهيم، بالإضافة إلى الأشياء ذات الدلالة الملموسة بحيث تكون وحيدة (غير متكررة) بواسطة معرف معين Identifier.

فى نموذج كينونة علاقة المطور EER تعريف تشييدات المخطط يتم بناؤها بإعطائها أسماء لا تتكرر. بالإضافة إلى أسماء الخصائص الخاصة بالنوع المعطى التى يجب أن تكون مميزة.

- التصنيف والتفريع Classification and Instantiation :

يتضمن التصنيف التبويب الشيئى المتشابه فى أنواع. وعلاقة الربط بين الشىء والنوع تكون علاقة ربط "عضو - أو" Is-Member Or. وعلى العكس من التصنيف يكون التفريع.

- مفهوم النوع الأصلى والنوع الفرعى Subclass and Superclass Concepts :

إن وقائع نوع كينونة معينة ربما يكون فئة جزئية من وقائع نوع كينونة أخرى. مثال ذلك: نوع كينونة "الطالب" STUDENT قد تكون نوعًا فرعيًا من نوع كينونة "شخص" PERSON. وعلاقة الربط بين نوعى الكينونتين تسمى "يكون عضواً" IS-A.

- وراثة الخاصية Attribute Inheritance :

الكينونة التى تكون عضواً لنوع فرعى ترث كل خصائص الكينونة كأنها عضو فى النوع الأصلى. والكينونة أيضًا سوف تورث كل وقائع قيم علاقات الربط لأنواع علاقة - الربط التى يساهم فيها النوع الأصلى .

- هرميات التعميم Generalization Hierarchies :

توجد هرميات الأنواع حيث يرتبط النوع الأصلى بعدد من الأنواع الفرعية التى تخصه بناء على المعالم التى تميزها. مثال ذلك: النوع الفرعى "شخص" PERSON قد يرتبط بالنوع الفرعى "طالب" STUDENT و"موظف" EMPLOYEE. ففى هذه الحالة، قد ينقسم النوع الفرعى "طالب" بناء على الوضع الدراسى إلى "طلاب خريجين" و"طلاب يدرسون". وأيضًا النوع الفرعى "موظف" قد ينقسم بناء على وقت العمل إلى "وقت - كامل" و"وقت - جزئى". ويتم تمثيل هرمية كل نوع بتجريد **التعميم** عن طريق التحرك تجاه قمة الهرم ، فى حين أن **التخصيص** عن طريق التحرك تجاه أسفل الهرم.

- قيود التخصيص والتعميم:

القيود التى تحكم الأنواع الأصلية عكس القيود التى تحكم الأنواع الفرعية. ويمكن أن تنمذج "كإجمالى" عكس "جزئى" ، و"كمساهمة إجبارية" عكس "المساهمة الاختيارية" . والتخصيص الإجمالى يصف القيد الذى فيه كل كينونة فى النوع الأصلى بأنها يجب أن تكون عضواً لنوع فرعى معين فى التخصيص، فى حين أن التخصيص الجزئى لا يسمح للكينونة أن تنتمى إلى أى أنواع فرعية.

التعميم فى النوع الأصلى عادة يكون إجمالياً: لأن النوع الأصلى يكون مقسماً إلى أنواع فرعية. لو أن دلالية علاقة ربط معينة هى أن كل واقعة تخص نوع كينونة معينة يجب أن يسهم فى علاقة الربط ، عندئذ نوع عضوية نوع الكينونة تصبح إجبارية فى تلك العلاقة ، وبخلاف ذلك فإن نوع العضوية يكون اختيارياً.

نماذج البيانات الوظيفية Functional Data Models :

تمثل نماذج البيانات الوظيفية نوعاً آخر لنماذج البيانات الدلالية. وتستخدم نماذج البيانات الوظيفية مفهوم "الوظيفة الرياضية" كبناء نمذجة أساسى. تستخدم هذه النماذج الكينونات والوظائف على الكينونات ككتل بنائية أساسية. فى حالة تقديم أى طلب للمعلومات يتم عرضه Visualized فى حدود استدعاء وظيفى باستخدام المعاملات arguments. وتملك وظائف المستوى الأول الكينونات كمعاملات، ولكن يمكن أن يتم بناء وظيفة داخل وظيفة. حيث يمثل نموذج "دابلكس" DAPLEX أحد النماذج البارزة فى هذه التصنيف. ومن ثم يوجد نوعان من الوظائف التى يتم تعريفها باستخدام الكينونات كمعاملات أحدهما: وظائف القيمة - الواحدة التى ترجع كينونة. وثانيهما: وظائف القيم - المتعددة التى ترجع مجموعة كينونات. يسمح باستخدام وظائف المعاملات المتعددة. وهى توفر معانى تقليدية لتمثل علاقات الربط متضمنة كينونات عديدة. على سبيل المثال: يمكن الأخذ فى الحسبان الرسم التخطيطى لنموذج كينونة - علاقة المطور EER فى الشكل رقم (٦-٢٧): حيث يمكن استخدامه لتوصيف بعض أنواع الكينونات وعلاقات الربط فى توصيفات تشابه نموذج "دابلكس" DAPLEX الوظائفى. أنواع الكينونات هى: "شخص" PERSON و"طالب" STUDENT و"المناهج الدراسية" COURSE و"الإدارة" DEPARTMENT و"الأقسام" SECTION .

PERSON ( ) → ENTITY

STUDENT ( ) → ENTITY

COURSE ( ) → ENTITY

SECTION ( ) → ENTITY

DEPARTMENT ( ) → ENTITY

تصف الأوامر السابقة وظائف أنواع الكينونات التى تسترجع كينونات تجريدية ، ومن ثم فإن هذه الأوامر تعرف أنواع الكينونات المناظرة. وأيضًا خاصية أى نوع كينونة يمكن توصيفها كوظيفة ذات معامل argument هو نوع الكينونة وذات نتيجة هى الكينونة القابلة للطباعة. فعلى سبيل المثال: التوصيفات الوظيفية التالية تصف خصائص نوع كينونة "شخص" PERSON كالآتى:

SSN(PERSON) → STRING

BDATE(PERSON) → STRING

SEX(PERSON) → CHAR

وبخصوص الخصائص المركبة مثل الاسم Name فى الشكل رقم (٦-٢٧) يتم توصيفهم ككينونات، ومن ثم توصيف محتويات هذه الخصائص كوظائف لخاصية الاسم Name كالآتى:

NAME ( ) → ENTITY

NAME (PERSON) → NAME

FNAME (NAME) → STRING

MNAME (NAME) → CHAR

LNAME (NAME) → STRING

ولتوصيف نوع علاقة الربط ، لا بد من تسمية نوع هذه العلاقة الربطية وتعريفها أيضًا كوظيفة. على سبيل المثال: نوعا علاقة الربط "يتخصص" Major و"قاصر على"

Minor اللتان تربطان الطلاب بأقسام هيئة التدريس يمكن أن تعرف كالآتى:

MAJOR (STUDENT) → DSPARTMENT

MINOR (STUDENT) → DSPARTMENT

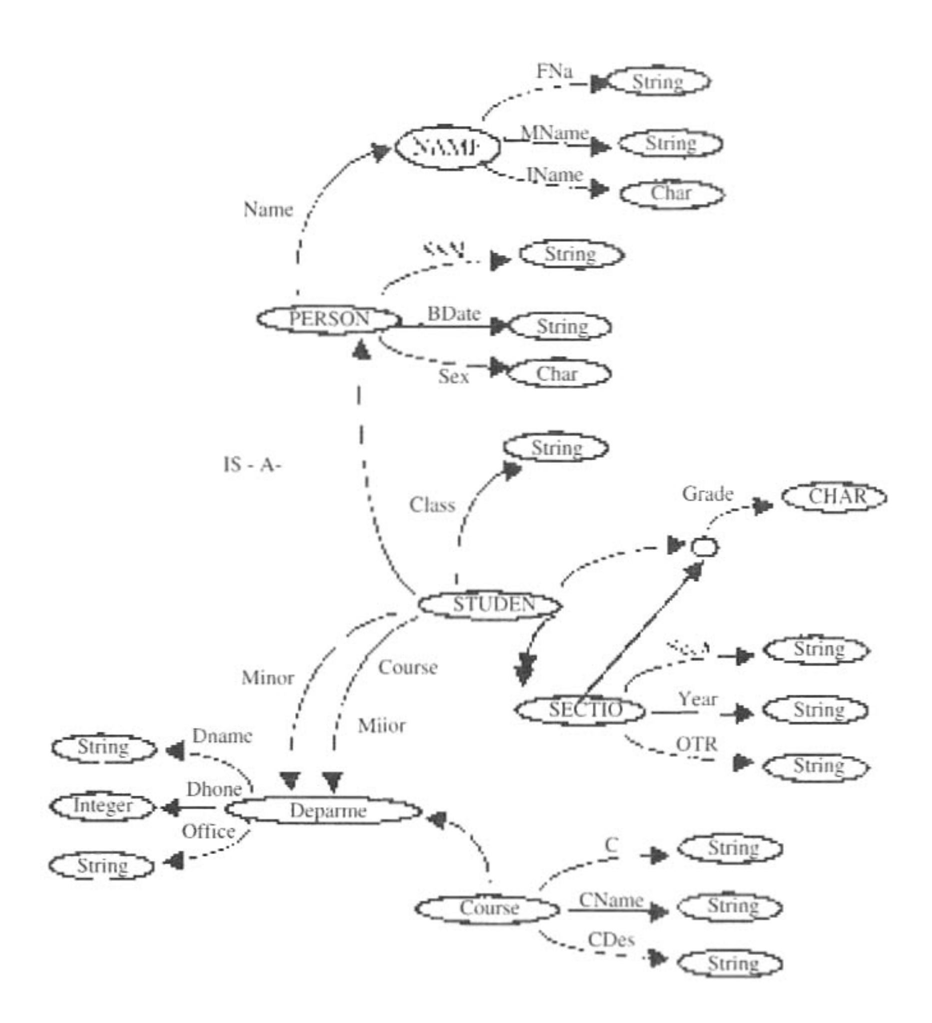
يصف هذا التوصيف تطبيق وظيفتى "يتخصص" Major أو"قاصر على" Minor لكينونة الطالب يتم استرجاعها كنتيجة كينونة لنوع كينونة "الإدارة" DEPARTMENT. ولتوصيف الدور العكسى لنفس نوع علاقة الربط يمكن أن تكتب كالآتى:

MAJORING-IN (DEPARTMENT) → STUDENT (INVERSE OF MAJOR)

MINORING-IN (DEPARTMENT) → STUDENT (INVERSE OF MINOR)

وعبارة Inverse تجعل هذه الوظائف عكس الوظائف الموصفة سابقًا، ومن ثم تصف نفس أنواع علاقات الربط فى اتجاه عكسى. وهذا توصيف لأنواع علاقات ربط واحد- لمتعدد (1: N). ويلاحظ أن هذا التدوين (→) يصف تطبيق وظيفتى "يتخصص" Major و"قاصرعلى" Minor على نوع كينونة "الإدارة" DEPARTMENT التى يمكن أن تسترجع مجموعة من الكينونات الخاصة بنوع "الطالب" STUDENT . وأيضًا يستخدم هذا التدوين لتوصيف الخصائص متعددة القيم. أما توصيف أنواع علاقات الربط متعدد - لمتعدد (M:N) فإنه يستخدم التدوين (↔) لكلا الاتجاهين وهكذا يبين الشكل رقم (٦-٣٩) تدوين الرسم التخطيطى لهذا المخطط المناظر لمخطط نموذج كينونة علاقة المطور EER [1] فى الشكل رقم (٦-٤٠).

شكل رقم (٦-٣٩) الرسم التخطيطى لتمثيل نموذج البيانات الوظيفى لنموذج كينونة علاقة المطور EER فى الشكل رقم (٦-٤٠)



ويمكن أيضًا توصيف علاقة الربط "يكون عضواً" IS-A والقيم المستنتجة، حيث إن وظيفة "يكون الطالب عضواً" Is-A-Student تصف علاقة الربط نوع أصلى/ فرعى فى مصطلحات نموذج كينونة علاقة المطور EER بين أنواع كينونات "شخص" PERSON و"الطالب" STUDENT كالآتى:

IS-A-PERSON (STUDENT) → PERSON

SSN (STUDENT) → SSN (IS-A-PERSON (STUDENT)

يمكن استنتاج بقية القيم من خلال وظيفة "يكون الطالب عضواً" Is-A-Student التى تصف علاقة الربط السابقة.

والمفهوم الوظيفى فى نموذج البيانات الوظيفى FDM هو مركب وظيفى. على سبيل المثال فإنه كتابة:

DNAME (OFFERING-DEPARTMENT (COURSE)

يتم تركيب وظيفتين، هما:

DANAMC . OFFERING-DEPARTMENT

ويلاحظ أن المركب الوظيفى هو مفهوم أساسى فى لغات الاستعلام الوظيفية. ويوضح ذلك بمثل بسيط يفترض استرجاع الأسماء الأخيرة Last Names لكل الطلاب المقيدين فى مادة "الرياضة" Math ، يمكن كتابه الاستعلام كالآتى:

RETR/EVE LNAME (NAME (IS-A-PERSON(MAJORING-IN (DE(PARTMENT)))

WHERE DNAME (DEPARTMENT) = "Math"

وهكذا فإن نماذج البيانات الوظيفية FDMs التى بدأت تتزايد وتنتشر بشكل واسع على الرغم من أنه لم يتم إنجازها بشكل جيد حتى الآن.

صفات نماذج البيانات الدلالية Characteristics Of SDMs :

يستخدم نموذج البيانات الدلالى لأغراض التصميم المفاهيمى: لذا يجب أن يجمع الصفات التالية [١٠]:

- **التعبيرية Expressionness :**

يجب أن يكون النموذج معبراً بشكل كافٍ لكى يمثل لك القدرة على التمييز بين مختلف أنواع البيانات والعلاقات بينها والقيود عليها.

- **البساطة Simplicity :**

يجب أن يكون النموذج بسيطًا على قدر الإمكان بالنسبة للمستخدم النهائى من جهة الاستعمال والفهم: لذا يجب دائمًا أن يرفق مع النموذج تدوين تخطيطى سهل.

- **التقليل Minimality :**

يجب أن يتكون النموذج من أقل عدد من المفاهيم الأساسية التى تكون مميزة ومستقلة فى المعنى.

- **الشكلانية Formality :**

ينبغى أن تكون مفاهيم النموذج معرفة شكليًا. كما أنها ينبغى أن تبين المعايير المؤكدة لصحة المخطط فى النموذج.

- **وحدانية التفسير Unique Interpretation :**

ينبغى أن توجد فكرة التفسير الدلالى الوحيد للمخطط المعطى. وهذا يضمن أن تكتمل الدلاليات ولا تحتوى على لبس فى التعريف لكل تركيبة النمذجة.

الهوامش :

1. [ELMASRI, 1989], Ramez Elmasri and Shamkant B. Navathe, **Fundamentals of Database Systems**, Benjamin/Cummings, Redwood City, Calif., 1989.

2. [Connolly 1996], Thomas. M. Connolly and Carolyn E. Begg, **Database Systems**, Addison-Wesley Publishing Co., Inc., 1996.

3. [Brathwaite 1991], Dr. Kenmore S. Brathwaite, **Relational Databases, Concepts, Design**, and Administration', McGraw-Hill Inc.New York,1991

4. [Shoval 1993], P. Shoval and Shreiber, 'Database Reverse Engineering From the Relational to the Binary Relationship Model', **Data and Khowledge Engineering**, 10 (3), 1993.

5. [Fred 1991], McFaden R. Fred and Hoffer A. Jeffrey, **Database Management**, Third Edition, the Benjamin / Cummings Publishing Co., Inc., 1991.

6. [Mcallister 1998], Mcallister, Andrew J. and Shorpe David, 'An Approach for Decomposing N-ary Data Relationships ', **Software-Practice and Experience**, Vol 28(2), Feb.,1998.

7. [MCLEOD, 1987], Hammer M. Mcleod, "Database Description with SDM: A Semantic Database Model', **ACM Trans. Database Syst.**, 19, 3, Sept.1987.

8. [LENZERINI, 1986], Batini C. Lenzerini and Navathe S.B., 'Acomparative Analysis of Methodologies for Database Schema Integration', **ACM comput. Surv.**, 18,4,Dec.,1986.

9. [DAVIES, 1992], P. Beynon Davies, 'Entity Models to Object Models: Object-Oriented Aalysis and Database Design', **Information and Software Technology**, Vol. 34, Number 4, April 1992.

10. [NAVATHE, 1992], Shamkant B. Navathe, 'Evolution of Data Modeling for Databases', **Communications of the ACM**, Vol. 35, No. 9, September 1992.

11. [Castano 1998], S. Castano, V. DE Antonellis, M.G. Fugini, and B. Pernici,'Conceptual Schema Analysis : Techniques and Applications ', **ACM Transaction on Database Systems**, Vol. 23, No. 3, September 1998.

12. [Rochfeld 1992], A. Rochfeld and P. Negros, 'Relationship of Relationships and Other Inter-Relationship Links in E-R Model', **Data and Knowledge Engineering**, 1992.

الفصل السابع

نماذج البيانات الشيئية الموجهة

مقدمة :

كل نوع لقاعدة بيانات عامة يجب أن يفرض السؤال التالى (١):

ما هياكل البيانات والعوامل المترابطة التى ينبغى أن تدعم النظام ؟

الإجابات المختلفة لهذا السؤال أدت إلى ظهور نماذج البيانات التقليدية الثلاثة التالية :

- الهرمية.
- الشبكية.
- العلاقية.

ولكن قد ظهر فى الآونة الأخيرة نوع آخر لنموذج قاعدة بيانات جديد ، وهو النموذج الشيئى الموجه. وتمثل نظم إدارة قواعد البيانات الشيئية الموجهة دمجًا للأسلوب الفنى لقواعد البيانات التقليدية والنموذج الشيئى . ويمكن لقواعد البيانات الشيئية الموجهة أن تعرض تحسينات معنوية على الأداء تفوق قواعد البيانات العلاقية التقليدية لتطبيقات معينة فى الحالات التى تتطلب تنفيذ ربط متعدد لجداول علاقية Relations كثيرة فإن قواعد البيانات الشيئية الموجهة يمكن أن تكون أكثر سرعة من قواعد البيانات العلاقية القابلة للمقارنة. القيود الأساسية على أنوا ع البيانات بالإضافة إلى النمذجة الشبكية المعقدة التى تؤدى إلى صعوبة استخدام للنموذج التشاورى للغة نظام البيانات CODASYL والنظم الهرمية لبعض التطبيقات مثل الوسائط المتعددة Multimedia ، والهندسية engineering وعلم والوراثة genetics وغيرها.

وسوف يتم استعراض الموضوعات التالية فى هذا الفصل:

أوجه التشابه بين نماذج البيانات الشيئية الموجهة والدلالية:

تندرج أوجه التشابه بين نماذج البيانات الشيئية الموجهة والدلالية فى حدود التجريد الهيكلى ومركبات النوع والقيمة.

مناطق الاختلاف بين نماذج البيانات الشيئية الموجهة والدلالية:

تتضمن مناطق الاختلاف بين نماذج البيانات الشيئية الموجهة والدلالية المعرفات والوراثة والكبسلة.

نموذج البيانات الشيئى الموجه OODM:

يحتفظ نموذج البيانات الشيئى الموجه بمناظرة مباشرة بين أشياء العالم الخارجى وقواعد البيانات ، بحيث لا تفقد الأشياء سلامتها أو هويتها.

قواعد البيانات الشيئية الموجهة:

تجمع قواعد البيانات الشيئية الموجهه كل من مفاهيم توجيه الشىء وإمكانيات قواعد البيانات ذات تقنية قاعدة البيانات الشيئية الموجهة.

اتجاهية الشىء:

تشمل اتجاهية الشىء : ثلاث مظاهر أساسية هى: أنواع البيانات التجريدية، الوراثة ، وهوية الشىء.

إمكانيات قواعد البيانات الشيئية الموجهة:

هناك العديد من إمكانيات قواعد البيانات بصفة عامة مثل: التزامية ، المعاملات ، المعالجة ، الاستعلام ، الأمن ، السلامة ، والاداء. ولكن قد تتميز قواعد البيانات الشيئية الموجهة عن غيرها من قواعد البيانات الأخرى بإمكانيتى الاستمرارية والإصرار. وسوف يقتصر التوضيح فى هذا الجزء على هاتين الإمكانيتين.

مزايا الطريقة الشيئية للشىء الموجه:

هناك العديد من المزايا للطريقة الشيئية مثل : التمديد ، التقييد بالمواصفات السلوكية ، مرونة التعريف ، وقوة النمذجة.

عيوب الطريقة الشيئية الموجهة :

وعلى الرغم من هذه المزايا إلا أن هناك بعض العيوب المرتبطة بهذه الطريقة مثل : فقدان الارتباط ، صلابة السلوك ، عدم اعتماد لغة الاستعلام على أساس علمى مثل نظرية الفئات التى سبق عرضها فى قواعد البيانات العلاقية.

لغة الأوبال :

تتكون لغة الأوبال من تعليمات خاصة بتعريف البيانات ومعالجتها. وتعد لغة الأوبال الأكثر أهمية فى معالجة البيانات، حيث تعتمد على نظم قواعد البيانات الشيئية الموجهة "جيم إستون GemStone" وهى من أقوى نظم قواعد البيانات الشيئية الموجهة. وقد تأثرت لغة الأوبال كثيراً بلغة Smalltalk فى تسهيلاتها لتعريف البيانات ومعالجتها.

أوجه التشابه بين نماذج البيانات الشيئية الموجهة والدلالية:

تتضمن قواعد البيانات الشيئية الموجهة معالم نماذج البيانات الدلالية والبرمجة الشيئية الموجهة . وتتشابه نماذج البيانات الشيئية الموجهة مع النماذج الدلالية فى حدود الآتى[٢] :

١- التجريد الهيكلى Structural Abstraction :

كلاهما يوفر التجريدات الهيكلية التى سبق ذكرها فى الفصل السابق . فى حين تعتبر الوراثة inheritance ضرورية فى كل النماذج الشيئية الموجهة بينما لا تعتبر كذلك فى كل نماذج البيانات الدلالية.

٢- مركبات النوع والقيمة Value and type Composition :

كلاهما يوفر مشيدات Constructors للنوع والقيمة التى تسمح لمصممى التطبيقات أن ينشئوا أعلى النماذج للقيم أو الأنواع الضخمة bulk - types مثل القوائم والمجموعات والحقائب والصفوف.

مناطق الاختلاف بين نماذج البيانات الشيئية الموجهة والدلالية:

المناطق التى تختلف بها نماذج البيانات الشيئية الموجهة عن نماذج البيانات الدلالية هى[٣]:

(١) معالجة المعرفات Treatment of Identifiers :

تعتمد نماذج البيانات الدلالية فى تركيب المعرفات أو المفاتيح على الخصائص أو الصفات الداخلية. فى حين تستعمل النماذج الشيئية الموجهة لمعرفات الخارجية للشىء (هوية الشىء) حيث إن المعرفات تحتفظ بعدم التغيير فى حين أن الأشياء ربما تفقد تغييراتها. هذه المعرفات تسمى "Surrogates" نواباً أو معرفات النظام.

(٢) معالجة الوراثة Treatment of inheritace :

تحقق النماذج الدلالية وراثة الخاصية فقط بين الأنواع الأصلية والأنواع الفرعية وهى محدودة. من ناحية أخرى توفرالنماذج الشيئية الموجهة كلاً من الوراثة الهيكلية والسلوكية ، وهذا يشير إلى وراثة الصفات الهيكلية والإجرائية أو البرمجية؛ ولذا فإن الوراثة بين الأنواع غير مرتبطة بنوع علاقة الربط التى يسمح بها فى النماذج الدلالية.

الكبسلة وإخفاء المعلومات Information-Hiding & Encapsulation :

فى النماذج الدلالية، البرامج (الطرق) Methods المكبسلة (المغلقة) داخل النوع الشيئى تسمح لمتغيراتها أو خصائصها بأن يتم تداولها، فى حين لا يوجد أسلوب لتداول هذه المتغيرات أو الخصائص مباشرة فى نموذج البيانات الشيئى الموجه OODM.

نموذج البيانات الشيئى الموجه OODM :

قد ظهر فى الآونة الأخيرة اتجاه جديد فى نمذجة البيانات ومعالجة قواعد البيانات فى هذا الاتجاه تعتبر قواعد البيانات مجموعة أشياء Objects، حيث إن كل شىء إما أن يمثل كينونة مادية، مفهوماً، فكرة، حدثًا، أو أى مظهر يخص تطبيق قاعدة البيانات. بينما فى نماذج البيانات التقليدية ذات السجل الموجه (الهرمية، الشبكية، العلاقية) كان ينظر للبيانات كمجموعة أنواع سجلات أو جداول علاقية ، كل منها له مجموعة سجلات

أو مجموعة قيم مرتبة مخزنة داخل ملف ولكن فى النظام الشيئى الموجه، يتم تمثيل أشياء العالم الحقيقى مباشرة بواسطة أشياء قواعد البيانات. وتكون هوية الشىء Object Identity محفوظة عبر معرف الشىء Object Identifer.

بينما الأنواع الأكثر تعقيداً لأشياء العالم الحقيقى يتم نمذجتها فى تطبيقات قواعد البيانات البازغة التى تستخدم نماذج البيانات التقليدية والتى تؤدى إلى بعثرة المعلومات على جداول علاقية أو أنواع سجلات كثيرة تقود إلى فقدان المناظرة المباشرة بين شئ العالم الحقيقى وتمثيله داخل قواعد البيانات. وأحد أهداف نمذجة الشىء الموجه هو أن يحتفظ بمناظرة مباشرة بين أشياء العالم الخارجى وقواعد البيانات بحيث إن الأشياء لا تفقد سلامتها أو هويتها ويمكن تعريفها وتداولها بسهولة.

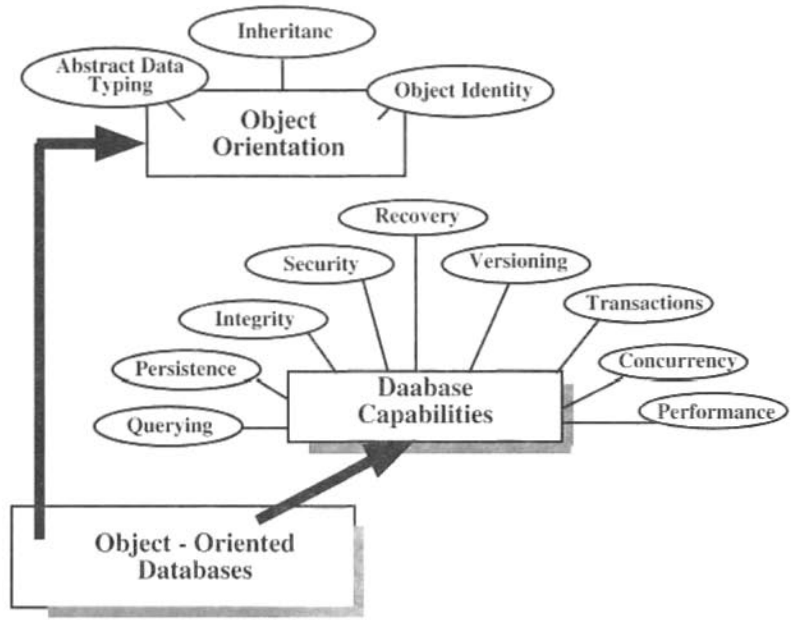
قواعد البيانات الشيئية الموجهة OODBs :

قواعد البيانات الشيئية الموجهة تجمع وتكمل مفاهيم توجيه الشىء لكى تشبع احتياجات الحاسب ليس فقط لتطبيقات قواعد البيانات المتقدمة ولكن أيضاً كقاسم مشترك يوضح كلاً من الشىء الموجه وإمكانيات قواعد البيانات ذات تقنية قاعدة البيانات الشيئية الموجهة. لذلك فإن التعريف الآتى يمكن استخدامة كإطار عام يميز سمات قواعد البيانات الشيئية الموجهة. وقواعد البيانات الشيئية الموجهة = اتجاهية الشىء + إمكانيات قواعد البيانات

Object-Oriented DB = Object Orientation + Database Capabilities.

ويوضح الشكل رقم (٧-١) النظام القوى الذى يطلق عليه قواعد البيانات الشيئية الموجهة التى تجمع بين فوائد ومفاهيم اتجاهية الشئ مع امكانيات قواعد البيانات[٤].

شكل رقم (٧-١) قاعدة البيانات الشيئية الموجهة



المعالم الضرورية لنظم قواعد البيانات الشيئية الموجهة:

يستخدم مصطلح نظم إدارة قواعد البيانات الشيئية الموجهة لتوصيف نوع من الأنظمة بالإمكانيات الآتية:

١. القدرة على تعريف الأشياء المعقدة Complex Objects.

وهذا يعنى القدرة على تعريف أنوا ع البيانات بالهياكل المتداخلة. على سبيل المثال: القيم المرتبة tuple يتم تشكيلها من أنوا ع أساسية مثل الأرقام الصحيحة، ويتم بناء الجدول العلاقى relation من مجموعات القيم المرتبة على شكل فئة بواسطة التجميعات.

٢. القدرة على تعريف الإجراءات المرتبطة بالأشياء.

وهذا يعنى أن تداولاً ينبغى أن يتم عبر الإجراءات المثبتة built-in. وهكذا يمكن تعريف الركام stack كنوع شيئى، ومن ثم فكل التداول الشيئى الركامى يكون عبر عمليتى الدفع Push والدس Pop اللتين ينبغى أن يتم تعريفها مسبقاً.

٣. القدرة على التمييز بين شيئين بنفس الصفات.

نموذج البيانات العلاقى لا يدعم فكرة هوية الشيء وراثيًا، بمعنى أنه لو كان هناك قيمتان مرتبتان two tuples متطابقتان فإنه لا يمكن تخزينهما فى أى جدول علاقى لأن الجدول العلاقى أساسًا هو فئة ، والفئات لا تسمح بالتكرار. بينما فى كل من النظم الشيئية الموجهة ونماذج البيانات التبحرية (الهرمية – الشبكية) تسمح بمثل هذه الدعم. وتركز نماذج البيانات الدلالية على تعريف هرمية الأشياء المعقدة وعلى وراثة المحتويات الهيكلية وعلاقات الربط عبر آليات التجميعات(مثل التخصيص والتعميم). فى حين تركز النماذج الشيئية الموجهة على التعريف وإمكانيات وراثة السلوك فى شكل إجراءات متداخلة داخل انوا ع الاشياء.

أولاً : اتجاهية الشيء Object Orientation -

اتجاهية الشئ هو نظام كل التطويقات (الإنجازات) التى اخترقت مجالات عديدة فى الحاسب متضمنة: اللغات ، واجهات تطبيقات المستفيد ، الذكاء الصناعى ، نظم التشغيل ، وقواعد البيانات وغيرها: لذلك فإن اتجاهية الشئ يمكن أن تعرف بشكل مطلق كنمذجة البرمجيات ونظم التطوير التى تجعل من السهل تركيب نظم معقدة من محتويات فردية. وأكثر المظاهر الاساسية الثلاثة لنموذج الشيء الموجه هى: أنوا ع البيانات التجريدية Abstract Data Types (ADTs) والوراثة Inheritance وهوية الشيء Object Indentity.

وتسهم كل من هذه المفاهيم فى هندسة البرمجيات وسمات النمذجة للنظم الشيئية الموجهة. وهكذا يمكن تعريف اتجاهية الشيء حسب المعادلة التالية[٤]:

اتجاهية الشيء = نوعية البيانات التجريدية + الوراثة + هوية الشيء

وفيما يلى شرح أكثر تفصيلاً لمفاهيم اتجاهية الشيء:

١- البيـانات التجريدية:

- التجريد Abstraction :

التجريد يشير إلى الصفات الأساسية للشيء والتى تميزه عن غيره من أنوا ع الأشياء الأخرى ويركز التجريد على السلوك behavior الضرورى للشيء دون تطبيقه implementation : ومن ثم فهو يركز على الصفات الضرورية للشيء بالنسبة لرؤية الشيء وفقًا لعلاقته الصحيحة وأهميتها النسبية للمشاهد. ولأنوا ع التجريدات من الأكثر إلى الأقل أهمية تتضمن الآتى[١] :

• تجريد الكينونة Entity abstract :

هو الشيء الذى يمثل نموذجًا مفيداً لمشكلة نطاق قيم الكينونة أو نطاق قيم حلولها.

• تجريد الحدث Action abstraction :

هو الشيء الذى يوفر فئة عامة للعمليات Operations ، كل منها يطبق نفس نوع الوظيفة (الدالة) .

• تجريد الآلة الافتراضى Virtual machine abstraction :

هو الشيء الذى يجمع العمليات معًا والتى يتم استخدامها بواسطة المستوى الأعلى للتحكم أو العمليات التى تستخدم مجموعة عمليات المستوى الأدنى.

• تجريد التماكن/التزامن Coincidental abstraction :

هو الشيء الذى يغلف مجموعة عمليات ليس لها علاقة بأى عملية أخرى.

أنواع البيانات Data types :

توفر كل لغات البرمجة دعمًا لأنوا ع البيانات ، ومثال ذلك : لغة البسكال تدعم أنواعًا أساسية مثل الأرقام الصحيحة integers ، الأرقام الحقيقية reals ، الحروف characters ، منشئ الأنوا ع type Constractor مثل المنظومات arrays والسجلات records. ويصف نوع البيانات كيفية تمثيل مجموعة من الأشياء ، فعلى سبيل المثال فإن نوعًا معينًا قد يتكون من الآتى:

اسـم المـادة : Name

تعريف رقم المادة : Number

تكلفة المـادة : Cost

وهكذا فإن كل من هذه الخصائص تسمى حقولاً fields أو ثقوبًا slots أو متغيرات واقعة instance variables لنوع بيانات المواد. وكل مادة لها قيم محددة لكل من هذه الخصائص. ويلاحظ أن اسم المادة Name هو سلسلة من الحروف String of characters وأن رقم المادة Number هو رقم صحيح وأن تكلفة المادة Cost هو رقم حقيقى Real ، وكلها أنواع بيانات أساسية. وحتى الآن مصطلح أنواع البيانات Data Types يستخدم لتوصيف مجموعة أشياء لها نفس التمثيل. بالإضافة إلى بعض العمليات التى ترتبط بأنواع البيانات، فعلى سبيل المثال : العمليات الحسابية على الأرقام الصحيحة والحقيقية ، عمليات لصق سلاسل الحروف Concatenate وعمليات استرجاع وتعديل تكلفة المادة. وهكذا فإن التعريف العام لنوع البيانات يمكن صياغته فى المعادلة التالية:

نوع البيانات = التمثيل representation + العمليات Operations

وفى اللغات التقليدية مثل البسكال والسى ، فإن عمليات نوع البيانات تتكون من منشئ نوع العمليات Constractors و نوع العمليات الأساسى base operations. ومنشئ أنواع البيانات الشائعة يعتنى السجلات والمنظومات والقوائم والفئات والتسلسلات. وهكذا فإن التعريف العام للعمليات يمكن صياغته طبقًا للمعادلة التالية:

العمليات Operations = منشئ نوع العمليات Constructor operations + نوع العمليات الأساسية base opweations.

على سبيل المثال: لو فرض أن سجل صنف معين له مجموعة من الخصائص. وأن أحد هذه الخصائص هو خاصية التكلفة Cost يتطلب تعديلاً نتيجة زيادة تكلفة الصنف ١٠٪. فإنه يجب تداول خاصية تكلفة الصنف Cost وتعديلها: ومن ثم فإن عملية التعديل يمكن تنفيذها باستخدام لغة البسكال كالآتى:

```
Item . Cost : = Item . Cost * 1.1;
```

أنواع البيانات التجريدية Abstract Data Types (ADTs) :

تستخدم أنواع البيانات لتوصيف مجموعة من الأشياء مع نفس علاقة الربط. وأكثر من ذلك نماذج أنواع البيانات التجريدية بأنواعها المختلفة فى تطبيقات قواعد البيانات الشيئية الموجهة، حيث إن واقعة كل نوع لها نظام: مجموعة رسائل يمكن أن تستجيب لها. ويوجد فصل واضح بين واجهة التطبيق الخارجية لنوع البيانات والتطبيق الداخلى باستخدام أنواع البيانات التجريدية، ومع ذلك التطبيق الداخلى لنوع البيانات التجريدية يكون مختفياً. ومن ثم التطبيقات البديلة يمكن أن تستعمل نفس نوع البيانات التجريدية بدون تغير واجهة تطبيقها. أنواع البيانات التجريدية تعرف فئات مكبسلة لأشياء متشابهة مع مجموعات مترابطة للعمليات؛ لذلك فإن أنواع البيانات التجريدية تصف الهيكل بالإضافة إلى سلوك الأشياء. توصيفات الهيكل تصف ما يشبهه الشىء. بينما تصف توصيفات السلوك ماهية الرسائل القابلة للتطبيق لكل شئ. وتخفى نوعية البيانات التجريدية التمثيل الداخلى للأشياء عن العالم الخارجى وتحمى الخطوات الداخلية التى تطبق سلوك الأشياء عن التطفلات الخارجية. وتوفر نوعية البيانات التجريدية والنموذج الحسابى شىء/ رسالة طريقة عمل أكثر "تفوضياً" delegatory للحساب ، حيث يرسل الشىء رسالة لشىء آخر الذى بدوره يحدد كيفية الاستجابة للرسالة. بفرض تعريف نوع البيانات التجريدية مندوب المبيعات SalesPerson فى مثال مكتب الذكاء:

التمثيل	Representation
الاسم	Name
العمر	Age
رقم التليفون	Tel No
مكان المكتب	Office loc
المرتب	Salary
العمولة	Commission

الحساب	Accounts
الحصة النسبية	Quota
الطلبيات	Orders
العمليات	**Operations**
إضافة حساب جديد	AddNew Acount
حذف أمر	Remove Order
تغيير الحصة النسبية	Change Quota
تغيير العمولة	Change Commission
إجمالى الحسابات	Total Accounts

ويتم اختيار هياكل البيانات الفعلية لتخزين تمثيل نوع البيانات التجريدية غير المرئية للمستخدمين وأيضًا الخطوات التى تستعمل لتطبيق كل من عمليات نوع البيانات التجريدية التى تكون أيضًا مكبسلة داخل نوع البيانات التجريدية. إن أحد معالم نوعية البيانات التجريدية هو إخفاء – المعلومات Information - hiding، بمعنى أن الأشياء لها واجهات تطبيق عامة Public. ومع ذلك التمثيل فإن التطبيق لهذه الواجهات يكون خاص Private. وهذا ما يبينه الشكل رقم (٧-٢). حيث إن الشكل رقم (٧-٢أ) يبين الواجهة العامة للنوع Class وتطبيقه ، ويتعامل مستخدمو النوع Class مع واجهة التطبيق (البرامج أو العمليات) وليس التطبيق للنوع، وهو ما يوضحة الشكل رقم (٧-٢ب) لنوع مندوب المبيعات Salesperson.

النوع Class والشىء OBJECT :

يعتبر النوع هو القالب Mold or templete يستعمله الحاسب لإنشاء الأشياء Objects. أما الشىء فهو واقعة النوع. ويعتبر الشىء واقعة لنوع Class واحد فقط. يجب تعريف النوع فى لغات البرمجة الشيئية الموجهة قبل إنشاء واقعة (الشىء) النوع[٥].

الرسائل والبرامج Messages & Methods :

يتم استخدام الرسائل messages كإصدار أمر للنوع أو للشيء بتنفيذ مهمة معينة؛ لذا يتم إرسال رسالة له. فعلى سبيل المثال: يمكن إرسال رسالة جديدة لنوع "حساب" Account لإنشاء واقعة حساب؛ ومن ثم يمكن إرسال رسالة لشيء الحساب لإيداع مقدار $١٠٠. وعلى النوع أو الشيء أن يعالج الرسالة التى يتم تسلمها من قبل البرنامج (الطريقة) الجارى على تعليمة التنفيذ ويسمى البرنامج المعرف للنوع باسم برنامج النوع Class Method والبرنامج المعرف للشيء بإسم برنامج الواقعة Instance Method والقيمة التى يتم تمريرها للشيء تسمى معامل الرسالة message argument[٥].

وتعريف النوع Class يتضمن الآتى:

١- اسم النوع Class name .

٢- العمليات الخارجية External Operations لمعالجة وقائع النوع Class Instances وهذه العمليات لها شيئ مستهدف وعدداً من المعاملات arguments . وتسمى عمليات واجهة التطبيق ببرامج (بطرق) النوع Class methods.

٣- التمثيل الداخلى Internal representation وهو الذى يستولى على قيم الحالات المختلفة لوقائع النوع.

عوامل واجهة التطبيق (methods) لنوع مندوب المبيعات SalesPerson تتضمن الآتى:

١- الاسم Name	إضافة حساب جديد AddNewAccount .
المعامل argnment	الحساب المخصص لمندوب المبيعات.
التأثير effect	إضافة وتخصيص حساب جديد لمندوب المبيعات.
٢- الاسم Name	إلغاء الطلب Remove Order .
المعامل argument	الطلب الذى لم يعد نشطاً.
التأثير effect	حذف الطلب

٣- الاسم Name الحسابات الإجمالية Total Accounts.

المعامل argument لا يوجـد

التأثير effect العودة بالرقم الإجمالى للحسابات المخصصة لمندوب المبيعات

شكل رقم (٧-٢) البرامج وتطبيق النوع الخاص لمندوب المبيعات

شكل رقم (٧-٢أ) البرامج والتطبيق

Method 1	Method 2	Method 3	مستخدمو النوع ينفذون البرامج من خلال الرسائل
Internal Implementation of the Methods			يصف النوع التطبيق اللازم لبرامجه (أى لواجهة تطبيقه)

شكل رقم (٧-٢ب) يوضح النوع الخاص لمندوب المبيعات

AddNew Account
Give Raise
Change Quota
Total Account

Class Sales Person the Methods

Name
Age
Account
Orders

Instance Variablred

AddNew Account NA
Accounts:= Accounts union {NA}

Methods Implenentations

Class Sales Persons Implementation

يجب أن يتضمن تعريف النوع التشفير Code الذى يطبق عوامل واجهة تطبيق النوع، بالإضافة إلى التمثيل الداخلى للأشياء (حالات الشىء) فى النوع. وقيم المتغيرات فى التمثيل الداخلى لوقائع النوع تخص الأشياء الفردية individual objects. فعلى سبيل المثال : التمثيل الداخلى لـ "أحمد" يتكون من توصيفة (الاسم،العنوان،...الخ) والحسابات والأوامر النشطة التى يتعامل معها . بعض القيم التى تصف "أحمد " قد تصف أشياء أخرى مثل : "أحمد" يشارك مكتبه مع "صلاح "؛ لذلك يعتبر المكتب قيمة مشتركة . وإن تجميع الفئة الكاملة لهذه القيم يستحوذ على حالة أحمد State of Ahmed لواقعة مندوب المبيعات. ومع ذلك فإن قيم المتغيرات فى التمثيل الداخلى مختلفة عن كل نوع واقعة، وكل الوقائع تشارك التشفيرات التى تطبق عوامل واجهة التطبيق. وعوامل واجهة التطبيق لها غرض متشابه لاستدعاء الإجراء procedure فى لغات البرمجة التقليدية.

وهكذا فإن قاعدة تشفير فردية تطبق مثل هذه العوامل مثل: إضافة حساب جديد Add New Account ، إلغاء الطلب Remove Order ، الحسابات الإجمالية Total Accounts. هذه العوامل Operators توضع دائمًا موضع التنفيذ داخل الشىء المستهدف Target Object كمعامل argument. والنظم الشيئية الموجهة توظف العمليات الدقيقة للأشياء المستهدفة بدون انتهاك لحالتها الداخلية Internal States.

مميزات نوعية البيانات التجريدية:

يمكن أن تتلخص مميزات نوعية البيانات التجريدية فيما يلى:

١- أنها تسمح بأفضل نمذجة مفاهيمية ولنمذجة العالم الحقيقى بتعزيز التمثيل والقدرة على الفهم وبتصنيف الأشياء المبينة على الهيكل والسلوك الشائع.

٢- أنها تعزز قوة النظام. فلو أن اللغة تسمح ضمنياً بتوصيف الأنوا ع لكل متغير، فإن نوعية البيانات التجريدية تسمح بمراجعة النوع لتجنب أخطاء النوع وقت المعالجة run-time. بالإضافة إلى مراجعة قيود السلامة على البيانات والعمليات التى تعزز صحة البرامج.

٣- أنها تعزز الأداء ، بجعل وقت الترجمة Compile-time أفضل ما يكون بمجرد معرفة أنوا ع الأشياء. كما أنها تسمح بأفضل سياسة تجميع لاستمرارية الأشياء.

٤- أنها تستحوذ على دلالية النوع بشكل ناجح ، بتجميعات نوعية البيانات التجريدية ، أو تمركزها والعمليات وتمثيل الخصائص.

٥- أنها تفصل التطبيق عن التوصيف وتسمح بتعديل وتعزيز التطبيق دون التأثير على واجهة التطبيق العامة لنوع البيانات التجريدية.

٦- أنها تسمح بالقدرة على التمديد extensibility للنظم التى لها محتويات برمجيات قابلة لإعادة الاستخدام بحيث يكون من السهل إنشاؤها والاحتفاظ بها.

٢- الوراثة Inheritance:

تمثل الوراثة علاقة ربط بين الأنواع التى يرث بواسطتها أى نوع كل التوصيف لنوع آخر بين الأنواع العامة أو جزءاً منه، وترث الوقائع كل السمات والبرامج للأنواع التى تنتمى إليها.

وللوراثة مميزات عديدة ، أنها تسمح لنوع معين أن يرث السلوك (العمليات ، البرامج ، ..إلخ) والتمثيل (المتغيرات ، الخصائص ، وهكذا) من الأنواع الموجودة. وتوريث السلوك يمكن من مشاركة تشغيل البرامج: ومن ثم يمنح القدرة على إعادة استخدامها بين البرامج الفرعية modules . وتوريث التمثيل يمكن من المشاركة الهيكل بين بيانات الأشياء. يتم إنجاز الوراثة بتخصيص أنواع موجودة، والأنواع Classes يمكن أن تكون مخصصة بوجود تمثيلها للمتغيرات أو عمليات السلوك[٤].

التخصيص هو تقنية أعلى - أسفل لتطوير تطبيقات قواعد البيانات الشيئية الموجهة . أما التعميم فهو المتمم للتخصيص ، ويستعمل تقنية أسفل - أعلى بإنشاء أنواع التعميم أو الأنواع الأصلية Superclasses/الأنواع فرعية Subclasses موجودة.

بالإضافة إلى نمذجة تطبيقات العالم الحقيقى بشكل مغلق على قدر الإمكان ، وتحاول اتجاهية الشىء أن تنجز برمجيات قابلة لإعادة الاستخدام وأيضًا برمجيات قابلة للتمديد. وتعد الوارثة قوة أخرى لمفهوم الشىء الموجهة الذى يوفر هذه الإمكانيات وتستطيع الوراثة التى لها جذور فى نماذج تمثيل المعرفة Knowledge-representation المستعملة فى الذكاء الاصطناعى، على بناء أنواع أشياء جديدة ووحدات برمجية

Software modules (Class) على قمة الهرم الموجود للوحدات البرمجية؛ مما يؤدى إلى تجنب إلى إعادة التصميم وإعادة التشفير من البداية. ويمكن أن تورث أنواع Classes جديدة كلاً من السلوك behavior (العمليات Operations البرامج (الطرق) methods ، ...إلخ) والتمثيل representation (لمتغيرات القيم Instance Variables ، والخصائص atlributes ، ...إلخ) من الأنواع الموجودة . ويمكن توريث السلوك من مشاركة التشفير ومن ثم إتاحة القدرة على إعادة الاستخدام بين وحدات البرمجيات . ويمكن توريث التمثيل من مشاركة الهيكل بين أشياء البيانات. وتجميع هذين النوعين للوراثة يوفر نمذجة قوية . وتوفر الوراثة أيضًا آلية طبيعية لتنطيم المعلومات ، حيث إنها تصنف الأشياء تصنيفًا علميًا فى تعريف جيد لهرمية الوراثة.

الوراثة Inheritence فى برمجة الشيء الموجه تستعمل آليه تسمى الوراثة يستخدم لتصميم كينونتين أو أكثر والتى تكون مختلفة ولكن يشارك فى معالم كثيرة شائعة. ويسمى النوع الشائع والنوع الأصلى والأنواع التى ترث منه بالأنواع الفرعية [٥].

أشكال الوراثة Facets of Inheritance :

تقدم الوراثة بعض التعقيدات Complexities خصوصًا عندما تتكامل مع مفاهيم الشيء الموجه الأخرى مثل الكبسلة encapsulation والنوعية typing ، الوضوح visibility وحالات الشيء Object States . وهناك ستة أشكال للوراثة التى تمييز معظم الطرق الفهمية المستخدمة بواسطة اللغات الشيئية الموجهة وهى[٤]:

١- الوراثة والنوعية الفرعية Inheritance & Subtyping :

فى معظم اللغات الشيئية الموجهة تكون الوراثة والنوعية الفرعية المستعملة قابلة للتبادل. ولغات قليلة توفر تشييدات مختلفة لدعم كل مفهوم. وهما مفهومان مدمجان فى آلية واحدة فى معظم لغات البرمجة. وبينما مصممو اللغات يرادفون استعمالاً بين مصطلحى نوع Class ومصطلح نوع type وهما مصطلحان مترادفان. هكذا أيضًا يرادفون استعمالاً بين مصطلح النوع الفرعى Subclass : الذى يشير إلى الوارثين inheritors ومصطلح النوع الفرعى Subtypes : مما يؤدى إلى معانٍ مختلفة فى لغات مختلفة. والنوع type هو مجموعة أشياء ومجموعة عمليات على الأشياء، والعناصر

elements الشائعة لمختلف الأنواع يمكن أن تكون مجردة لتشكل هرميات النوع الفرعى Subtype . وبشكل عام تتعامل الوراثة مع التطبيق، والنوعية الفرعية هى علاقات الربط الدلالية بين أنواع الأشياء. بالوراثة يمكن للمبرمجين تشييد نظم أكثر تخصصاً من هرميات النوع الموجود. إنها الآلية التى تسمح للوحدات البرمجية أن تشير إلى الوحدات البرمجية وتعيد استخدامها. أما فى اطار الشىء الموجهة فإن النوعية الفرعية يتم تحليلها مقترناً بنوعية البيانات التجريدية. وهكذا لو كان هناك نوعان من نوعية البيانات التجريدية. الأول نوعية البيانات التجريدية ADT1 وهو نوع فرعى من نوعية البيانات التجريدية الآخر ADT 2 فإن:

أ – هيكل ADT1 يجب أن يكون نوعاً فرعياً Subtype لهيكل ADT2.

ب – سلوك ADT1 يجب أن يعمل وفقاً لسلوك ADT2.

وتعريف السلوك للنوعية الفرعية يكون مستقلاً عن التطبيق وينظر للنوعية الفرعية كهرمية السلوك . بينما من ناحية أخرى ينظر إلى الوراثة كهرمية تطبيق Imple- mentation .

٢– وضوح المتغيرات الموروثة والبرامج Visibility of inherited variables and methods :

تسمح بعض اللغات الشيئية الموجهة بالمعالجة المباشرة لمتغيرات الوقائع Instance Variables. وبعض اللغات الأخرى تميز بين متغيرات الوقائع العامة والخاصة . وبالوراثة توجد ثلاثة بدائل تسمى النوع الفرعى – المرئى Subclass-Visible.

أ– وراثة النوع Class inheritance:

فى معظم اللغات الشيئية الموجهة يتم إدماج الوراثة فى اللغات الشيئية الموجهة خلال وراثة النوع Class inheritance. وتطبق الأنواع Classes الفئات المكبسلة للأشياء التى تعرض نفس السلوك (أى نفس أنواع البيانات التجريدية) وفى معظم هذه اللغات مثل لغة ++C يمكن أن تورث الأنواع كلاً من الطرق Methods (السلوك behavior) ومتغيرات الوقائع instance variables (الهيكل Structure) من الأنواع الأصلية superclasses.

وبجانب توفير أداة قوية لتنظيم المعلومات ، فإن المساهمة الأكثر أهمية هى مشاركة التشفير أو القدرة على إعادة استخدام التشفير. ويمكن توضيح القدرة على إعادة استخدام التشفير والاستيلاء على وراثة النوع Class inheritance فى النظم الشيئية الموجهة باستخدام المثال الذى يبين الأنواع classes المتمركزة فى رأس الشجرة فى الشكل رقم (٧-٣) والممثلة لنوع موظفى المكتب Office Worker التى يمكن أن تكون "مخصصة" Specialized بالإضافة إلى هرمية الوراثة والأنواع هى:

السكرتارية Secretary، مندوبو المبيعات Sales Persons الذين هم موظفو المكتب ، ويعتبر مديرو المبيعات Sales Managers أكثر تخصيصاً لمندوبى المبيعات والمديرون المباشرون تخصيصاً لمديرى المبيعات.

وتمثل الأنواع Classes: المطورين Developer، السكرتارية Secretary. ومندوبو المبيعات هم أنواع فرعية Subclasses لموظفى المكتب Office Worker، ويمثل نوع موظفى المكتب Office Woeker النوع الأصلى Supperclass. وتكون العلاقة بين النوع الفرعى والنوع الأصلى هى علاقة انتقالية transitive.

شكل رقم (٧-٣) يوضح هرمية الوراثة لموظفى المكتب

ويتبين مما سبق ذكره أن وراثة النوع Class Inheritance لها مظهران :

هيكلية Structure :

وفيه وقائع النوع مثل مندوب المبيعات Sales Person التى هى نوع فرعى لنوع موظفى المكتب Office Worker لها قيم متغيرات وقائع مورثة من نوع موظفى المكتب Office Worker مثل: الاسم Name، العنوان Address والمرتب Salary وهكذا.

السلوك Behavior :

توجد برامج للنوع مثل: تراكم الإجازات Accumulated Vacation، زيادة الرواتب GiveRaise، تغيير العناوين Change Address التى يتم توريثها بواسطة أنواعه الفرعية مثل مندوب المبيعات SalasPerson والسكرتارية Secretary. والنتيجة هى أنه يمكن للرسالة أن ترسل مع محدد الاختيار (GiveRaise Selector) إلى الواقعة "هالة" للسكرتارية؛ لكى تنفذ برنامج زيادة الرواتب Give Raise لموظفى المكتب Office Worker مع الواقعة "هالة" كشىء مستهدف.

ب - توريث البرامج (الطرق) Inheriting Methods :

مما سبق بيانه؛ فإن النوع Class يعرف كل من الهيكل Stracture والسلوك be-havior لمجموعة أشياء. ويتم توصيف السلوك فى البرامج المرتبطة مع وقائع النوع. والبرامج هى العمليات التى يمكن أن تسترجع أو تعدل حالة الشىء "Object State". ويتم تخزين حالة الشىء فى متغيراته.

فى هرمية الوراثة، يتم توريث البرامج Methods والمعرفة للنوع بواسطة أنواعه الفرعية. وهكذا البرامج المورثة هى جزء من واجهة التطبيق التى تعالج وقائع النوع الفرعى. فعلى سبيل المثال وثيقة النص Text decument ووثيقة الرسم Image decument فكلاهما يرث من نوع الوثيقة Decument Class التى لها العديد من البرامج مثل:

فتح Open

إغلاق Close

حفظ Save

حفظ باسم Save as

وهذه البرامج يتم توريثها بواسطة كل من وثيقة النص ووثيقة رسم. في النوع الفرعى لوثيقة نص فإن البرامج المخصصة لتحرير الوثيقة أو تعديل الخط وحجم الحرف وهكذا بالنسبة للنص الذى يتكون من مجموعة من حروف تكون معرفة. فإنه ليس من الضرورى أن تكون هذه البرامج معرفة لوثيقة واحدة بعينها بل يمكن تطبيقها لكل الوثائق النصية. وكذلك وثيقة رسم لها برامج محددة مثل التكبير والاستدارة التى يمكن تطبيقها للوثائق الرسم.

وكذلك بالنسبة لتعريفات متغيرات الوقائع ، فإن مجموعة من البرامج التى يمكن تطبيقها لواقعة النوع هى اتحاد لكل البرامج التى تعرف الأسلاف العليا ancestors للنوع بالإضافة إلى البرامج التى تعرف تعريف النوع.

٣- الوراثة والكبسلة Inheritance & Encapsulation :

رؤية متغيرات الوقائع تنتهك إخفاء المعلومات والتى تتضمنه كبسلة النوع. فى حقيقة الأمر لو كانت متغيرات الوقائع للأنواع الأصلية يتم تداولها بشكل مباشر، فإنه يوجد تضارب بين الوراثة والكبسلة. بل أكثر من ذلك فإن الكبسلة يمكن أن تستعمل لدعم بعض الوظائف الوراثية. ومع ذلك تكون الوراثة مباشرة بشكل أكثر وآليتها طبيعية لمشاركة التشفير والهيكلة.

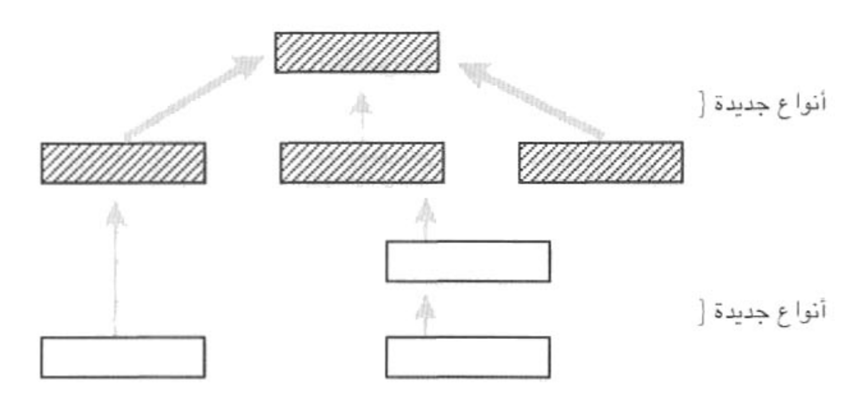
٤- التخصيص والتعميم Specidization & Generalization :

يتم إنجاز الوراثة بتخصص الأنواع الموجودة . والأنواع يمكن أن تخصص بامتداد تمثيلها (من خلال متغيرات الوقائع) أو السلوك (من خلال العمليات). ويمكن تخصيص الأنواع Classes بحصر التمثيل representation أو العمليات Operations للأنواع الموجودة. ومعظم النظم الشيئية الموجهة الموجودة تسمح بتطوير التطبيقات بواسطة تخصيص المحتويات الموجودة التى هى فى معظم الأحوال أنواع Classes خاصة بهذه التطبيقات. ويتم تحديد التطبيقات بإنشاء أنواع فرعية Subelasses للأنواع الموجودة. ويوضح الشكل (٧-١٤أ) والذى يشمل نوعين أحدهما الأنواع القديمة والمميزة بإطار به خطوط مظللة والأخرى هى الأنواع ذات الإطار الأبيض التى تم إنشاؤها بتوريث الهيكل والسلوك من الأنواع القديمة. ولذا فإن التخصيص هو تقنية أعلى- أسفل لتطوير البرمجيات وهو ما سبق توضيحه فى الفصل الخاص بنماذج البيانات الدلالية.

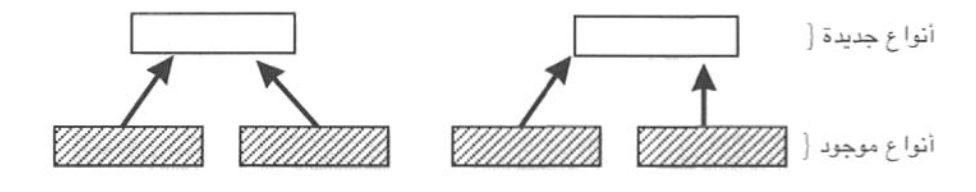
وعامة يتم البدء بهرمية النوع أى بالأنواع الأصلية فى مستوى القمة، ثم تمديدها بإنشاء الأنواع الفرعية التى تشكل أوراقًا Leaves للشجرة أو قاعدة للهرم. وتخصيص نوع موجود يمكن أن يتم بإضافة متغيرات الوقائع وحصر متغيرات الوقائع الموجودة وإضافة البرامج، وتجاوز البرامج الموجودة أو تجاهلها وهكذا. أما التعميم فهو المتمم للتخصيص. ويستخدم تقنية أسفل - أعلى لإنشاء الأنواع والتى تمثل الأنواع الأصلية للأنواع الموجودة . ويوضح الشكل رقم (٧-٤ب) والذى يبين الأنواع الجديدة فى قمة الهرم والتى يتم إنشاؤها باستخلاص الهيكل الشائع والذى يشمل متغيرات الوقائع والبرامج الموجودة من الأنواع الموجودة والتى تكون أسفل الهرم .

شكل رقم (٧-٤) التخصيص والتعميم

شكل رقم (٧-٤أ) الأنواع الجديدة هى تخصيص للأنواع الموجودة



شكل رقم (٧-٤ب) الأنواع الجديدة هى تعميم للأنواع الموجودة



٥- وراثة الشيء Object Inheritance :

تدعم معظم اللغات الشيئية الموجهة وراثة النوع Class، أى القدرة لنوع معين بوراثة التمثيل والبرامج من نوع آخر. والطريقة البديلة هى أن يسمح للأشياء Objects أن يتم وراثتها من شيء آخر. ووراثة الشيء تسمح لشيء أن يرث حالة State شيء آخر. بعض النماذج أيضًا تسمح بدمج عمليات مع الأشياء وتستعمل فقط لوراثة الشيء لتنظيم حيزه الشيء وتسمى نظم النماذج الأولية Proto type systems. فى هذه النماذج ترسل الأشياء رسائل لأشياء أخرى ومن ثم توريث البرامج أو القيم المخزنة فى أشياء أخرى.

٦- الوراثة المتعددة Multiple inheritance :

فى العديد من الحالات يكون من المرغوب أن تتم الوراثة من أكثر من نوع واحد وتسمى الوراثة المتعددة . عندما يقوم نوع بالوراثة من أكثر من أب واحد تكون هناك إمكانية للتضارب قد تنشأ بواسطة البرامج (الطرق) Methods أو متغيرات الوقائع Instance Variables مع نفس الاسم تؤدى إلى توريث دلاليات مختلفة أو غير مرتبطة من أنواع أصلية مختلفة. وبالوراثة المتعددة يمكن أن يتم جمع العديد من الأنواع الموجودة لكى تنتج أنواع تستخدم الأنواع الأصلية المتعددة فى مختلف الطرق ومختلف الوظائف. بينما فى الوراثة الفردية يكون هرم وراثة النوع شجرة بنوع أكثر عمومية فى أصل الشجرة ، وهرم وراثة النوع للوراثة المتعددة له نوع يمكن أن يكون له أكثر من سلف فى نفس الوقت Predecessor ويصبح الرسم حلقة غير مغلقة مباشرة Directed Acyclic Graph (DAG)

ويتبين مما سبق أن مجموعة متغيرات الوقائع لنوع فرعى هى اتحاد Union لمتغيرات الوقائع للنوع الأصلى المباشر بالإضافة إلى متغيرات الوقائع المعرفة من قبل فى النوع الفرعى. وأيضًا مجموعة البرامج للنوع الفرعى هى اتحاد لبرامج النوع الأصلى المباشر بالإضافة إلى البرامج المعرفة مسبقًا فى النوع الفرعى. ومع ذلك البرامج التى يحتويها النوع الفرعى يمكن أن تتجاوز (أو تتجاهل) البرامج فى النوع الأصلى.

مميـــزات الوراثة Advantages of inheritance :

توفر الوراثة العديد من المزايا لنمذجة مكاتب الذكاء، وهذه المزايا هى:

١- أنها تعرض نموذجًا طبيعيًا لتنظيم المعلومات . على سبيل المثال الوراثة تستحوذ مباشرة على الحقيقة أن مديرى المبيعات هم أيضًا مندوبو مبيعات.

٢- أنها تسمح بالتشفير والتمثيل للمشاركة بتخفيض الفاقد لتنظيم مكاتب الذكاء.

٣- أنها تسمح بأنوا ع وأشياء جديدة ليتم تعريفها فى قمة الهرميات الموجودة أفضل من تعريفها من العدم؛ مما يزيد من المرونة والقدرة على التمديد لهرميات نوع مكتب الذكاء.

٢ - هوية الشىء Object Identity :

الهوية هى صفة لشىء التى تمييزه عن كل الأشياء الأخرى فى التطبيق . ففى لغات البرمجة يتم تحقيق الهوية خلال عناوين الذاكرة memory address. أما فى قواعد البيانات تكون الهوية محققة خلال مفاتيح التعريف Identifier Keys. وهوية الشىء لها العديد من المزايا، من حيث الأشياء التى يمكن أن تحتويها أو الأشياء الأخرى التى تشير إليها. هوية الشىء توضح وتطور وتمتد إلى مؤشرات خيالية notions فى لغات البرمجة التقليدية، والمفاتيح الخارجية فى قواعد البيانات وأسماء الملفات فى نظم التشغيل. وباستعمال هوية الشىء فإن الأشياء يمكن أن تحتوى على أشياء أخرى أو تشير إلى ذلك. وهذا يؤدى إلى تجاهل الحاجة إلى استعمال متغيرات الأسماء التى ليس لها دعم لهوية الشىء، ولكنها تؤدى إلى بعض محدوديات الممارسة العملية. وأحد هذه المحدوديات هى فردية الشىء التى قد تؤدى إلى تداوله بطرق مختلفة. وهكذا قد يرتبط الشىء بمتغيرات مختلفة ليس لها طريقة للكشف سوى الإشارة إلى نفس الشىء. وهذه المحدوديات فى لغات البرمجة التقليدية قد تمنع استعمال هوية الشىء . على سبيل المثال: تم تعريف رجل المبيعات باسم P1 وقد تميزه كموظف ومدير للمبيعات حقق أعلى مبيعات فى شوال ١٤١٣هـ . نفس رجل المبيعات ارتبط باسم آخر مختلف هو P2 وقد تميز كرجل مبيعات له ثلاث رحلات خارج المملكة العربية السعودية أثناء عام ١٤١٣هـ. وبهذا الافتراض فإن P1 ، P2 يمكن أن يرتبطوا بأشياء ليس لها

مؤشرات Pointers . ولغات البرمجة التقليدية لا توفر شروطاً للارتباط مثل مطابقة الأشياء بشكل مباشر . وهذا على النقيض من اللغات الشيئية الموجهة التى توفر اختبار هوية بسيطة مع التعبير X = y ، والتى تختلف عن اختيار التكافؤ X = y . ويراجع اختيار الهوية ما إذا كان الشيئان متطابقين فى محتوياتهما أم لا.

١- مفاتيح المعرفات Identifer Keys :

وهذه طريقة أخرى لتعريف الأشياء باستعمال المفاتيح التى لا تتكرر أو مفاتيح المعرفات ، وهذه الآلية شائعة الاستعمال فى معظم قواعد البيانات وخاصة قواعد البيانات العلاقية التى سبق التطرق لها بإسهاب فى الفصل الرابع.

ولكن هناك ثلاث عقبات تنتج من استعمال مفاتيح المعرفات لهوية الشيء:

أ- تعديل مفاتيح المعرفات :

إحدى هذه العقبات هو أن مفاتيح المعرفات لا يمكن تغيرها حتى لو تم تعريفها بواسطة المستخدم.

ب- اللاتماثل Uniformity :

والمصدر الرئيسى للاتماثل هو أن مفاتيح المعرفات فى الجداول العلاقية relations المختلفة لها أنوا ع مختلفة أو تجميعات لخصائص مختلفة.

ج- الربط غير الطبيعى Unnatural joins :

والعقبة الثالثة هى أن الربط بمفاتيح المعرفات تستعمل فى الاسترجا ع بدلاً من الاسترجا ع بالشيء مباشرة الذى يعتبر أبسط ما يكون كما فى لغات الأوبال OPAL والفاد FAD والجيم GEM.

٢ - العناصر الثلاثة : النوع - الحالة - الهوية Type-State-Identity-Trichotomy :

إن النوع class - فى الأساس - يطبق نوعاً type يوصف كلاً من الهيكل والسلوك لوقائعه. ويتم استحواذ الهيكل فى متغيرات الوقائع بينما يستحوذ السلوك فى البرامج القابلة لتطبيق الوقائع.

وقيم متغيرات الوقائع للشيء تنشئ حالة State الشيء. بمعنى آخر فإن قيمة كل متغير هي الشيء ولتوضيح علاقة الربط بين الحالة State والنوع الأساسي type نستعرض المثال الآتي:

بفرض ان كل موظفي المكتب OfficeWorker له متغيرات مثل العنوان Address والاسم Name والعمر Age لها الأنواع الأساسية types الآتية:

Name :NAME

AGE : INTEGER

ADDRESS : ADDRESS

حيث إن الاسم ADRESS والعنوان INTERGER، والرقم الصحيح NAME . هي أسماء أنواع Classes، ونوع Class الاسم NAME يحتوي على المتغيرات الآتية:

LName :String of char .

FName :String of char .

ولذلك فإن كل واقعة لموظف المكتب OfficeWorker تحتوي على العديد من الوقائع التي تمثل قيم هذه المتغيرات فعلى سبيل المثال:

واقعة الاسم Name هي قيمة الاسم.

واقعة الأرقام الصحيحة Integer هي قيمة العمر AGE.

واقعة العنوان ADDRESS هي قيمة العنوان ADDRESS.

ومن ثم فإن:

١- الشيء هو واقعة للنوع Class (أي نوعه الأساسي its type).

٢- الشيء له حالة State هي قيمة متغيراته.

بالإضافة إلى أن لكل شيء هوية identity مثبتة built-in ومستقلة عن نوعه Class وحالته State . وهوية الشيء تولد عند إنشائه وهي دائمة بينما حالة الشيء (قيم

متغيراته) يمكن أن تتغير عفويًا ، وهكذا فإن عنوان موظف المكتب OfficeWorker يمكن أن يتغير ولكن هويته تظل ثابتة كما هى. والنظم الشيئية الموجهة تدعم الهوية المثبتة بقوة على الرغم من أنها تسمح للشىء أن ينفذ تعديلات هيكلية (أى تعديل لنوعه Class) بدون أى تغيير لهويته. حالة الشىء تنشأ من قيم أساسية مثل الأرقام الصحيحة integers والحروف characters والسلاسل الحرفية strings والأرقام ذات النقطة العائمة Floating-point. وتستعمل القيم الأساسية للمتغيرات بدون هوية أو مراجع للشىء ، ويتم مشاركة الأشياء باستعمال أحد الحلين الآتيين:

١- الحل الأول هو تكرار الشىء والذى يؤدى إلى ضياع كل من المساحة التخزينية والاحتفاظ بتوافقية القيم .

٢- الحل الثانى هو استعمال "مفتاح المعرف" كما هو الحال فى نظم قواعد البيانات العلاقية.

وهوية الشىء لا تتطلب حلول التكرار الزائد ولا مفتاح المعرف. بل استعمال الهوية تسمح بالمعرف المنطقى (المؤشر Pointer) لكى يرتبط مع كل شىء فى النظام.

٣- تمثل حيز الشىء Object spaces repersentation :

يبنى حيز الشىء على قمة الأشياء الأساسية Base Objects. ونوع الشىء الأساسى الأكثر انتشارًا هو العدد الصحيح integer. أما أنواع الشىء الأساسية الأخرى هى الأرقام ذات النقطة العائمة، الحروف والمنطقيات boolean. والأشياء التى هى وقائع لهذه الأنواع عادة ليس لديها متغيرات. وهم مثبتون فى أنواع الأشياء التى يتم تدعيمها بواسطة النظام ضمنيًا. ومعظم النظم الشيئية الموجهة تخصص هوية للأشياء الأساسية. وهناك كم لا حصر له من ارتباط المعرفات مثل:

أ- يرتبط المعرف مع كل شىء لا أساسى.

ب- يرتبط المعرف بالشىء وقت إنشاء الشىء ويظلان مرتبطين بغض النظر عن تعديلات الحالة state التى type يتم تنفيذها بواسطة الشىء أو النوع الأساسى.

ج- يمكن ان يرتبط كل معرف مع شيء واحد فقط بمعنى أنه لو وجد معرف فى النظام لا بد وان يرتبط مع شيء. ويظل الشيء مرتبطًا بالمعرف فترة حياته. ولذا كل شيء له ثلاث صفات:

- الشيء هو واقعة لنوع ، وهذا يشير إلى النوع الشيء Object type.
- الشيء له هوية ، وهذا يشير إلى المعرف المرتبط بالشيء.
- الشيء له حالة ، وحالته هى قيمة متغيره.

ولو فرض بشكل أكثر تحديداً أن $A_1..A_n$ هى متغيرات للشيء O ، وحالة الشيء هى:

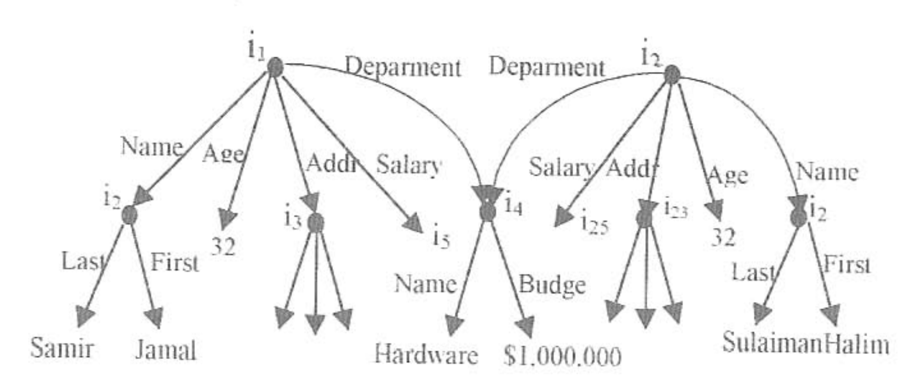
$$
\begin{array}{ccc}
A_1 & : & I_1 \\
A_2 & : & I_2 \\
. & : & . \\
A_n & : & I_n
\end{array}
$$

حيث إن I_j إما أن يكون معرف الشيء أو شيئًا أساسيًا.

ويوضح الشكل رقم (٧-٥) ، (٧-٦) التمثيل بالرسم لتلك للأشياء . ففى شكل رقم (٧-٦) ترتبط المعرفات مع الأشياء اللاأساسية، حيث كل شيء يوضع بإطار Frame لشكل مستطيلى. ويلاحظ أن قيمة المعرف لمتغير الإدارة Department لإدارة Jamal Samir هو شيء لا أساس. وكذلك قيمة المعرف لمتغير الإدارة Department لإدارة Sulaiman Halim. وهذا يعنى أن Jamal . Halim يشاركان نفس قيمة الادارة Department.

ويتتبع الشكل رقم (٧-٥) تمثيل نموذج الفئة set والقيمة المرتبة tuple. وأن كل شيء يعنون بالمعرف الخاص به ولكل متغير (خصائص فى نماذج الفئة واالقيمة المرتبة) يعنون ويوصل مباشرة بوصلة arc من الشيء إلى قيمة الشيء. والعنوان هو اسم المتغير والمستهدف هو قيمة المتغير. ويبين الشكل رقم (٧-٥) حيز الشيء فى شكل حلقة غير مغلقة مباشرة DAG. فى حين يبين الشكل رقم (٧-٦) التمثيل البديل باستخدام الأشكال المستطيلية.

شكل رقم (٧–٥) يتتبع تمثيل الشيء بالرسم



شكل رقم (٧–٦) يتتبع تمثيل الشيء بإطار مستطيلى

Obiect	i_1
Name	i_2
Age	32
Address	i_3
Salary	$ 32,000
Department	i_4

Obiect	i_{21}
Name	i_{22}
Age	32
Address	i_{23}
Salary	$ 34,000
Department	i_4
....	

Obiect	i_2
Last	Samir
First	Jamal

Obiect	i_3
Street#	42
St Name	K. Saud
City	Dammam
State	Eastern
Zip	31141

Obiect	i_{22}
Last	Sulainan
First	Halim

Obiect	i_4
Name	Hardware
Budget	$ 1,000,000

مما سبق يتضح أن لغات البرمجة تستخدم مؤشرات العنوان الافتراضى لكى تنجز القدرة على مرجعية الشىء object-referencing ، ولكى تسمح للمتغيرات أن تشير إلى نفس الشىء من مصادر متعددة . وفى الحقيقة أن المؤشرات أو عناوين الذاكرة الافتراضية يمكن أن تستعمل لتطبيق هوية الشىء. والاختلاف الأساسى بين هوية الشىء والعناوين الافتراضية أو المؤشرات هو أن الهوية مفهوم دلالى Semantic Concept مرتبط بالأشياء فى حين تمثل العناوين فى أماكن الذاكرة.

٤- العمليات المرتبطة بالهوية Operations With identity :

الهوية هى صفة الشىء التى تميزه عن غيره من الأشياء الأخرى فى البيئة الحسابية. والعناصر الثلاثة النوع - الحالة - الهوية تتضمن العديد من العمليات المرتبطة بهوية الشىء. ويمكن تصنيف العمليات كالآتى[٤]:

- العمليات الخاصة بشروط التساوى.
- العمليات الخاصة بالنسخ.
- العمليات الخاصة بالدمج والتبديل.

(أ) العمليات الخاصة بشروط التساوى Equality Predicates :

هناك ثلاث حالات من التساوى هى:

- التطابق وفيه يتم التأكد مما إذا كان الشيئان متطابقين أم لا.
- التساوى السطحى وفيه يتم التحرك إلى عمق مستوى واحد وتقارن قيم متغيرات الهوية أو عناصر الشىء المناظرة.
- التساوى العمقى وفيه تقارن المحتويات المناظرة للأشياء الأساسية.

وفيما يلى شرح مبسط لكل حالة من الحالات الثلاثة السابقة:

التطابق Identical :

وفيه يتم التأكد مما إذا كانت هويات الأشياء متطابقة أم لا. وبدلاليات هوية الشىء لو كانت متساوية فإن الأشياء أيضاً تكون متساوية.

ويبين الشكل رقم (٧-٧) ثلاث وقائع للشخص PERSON ، لكل واقعة اسم NAME وعمر AGE وعنوان ADDRESS. ومنه يبين أن :

O1. ADDRESS= = O2. ADDRESS

شرط صحيح أى أن عنوان Magdi ، Maria هما نفس الشيء المتطابق.

ولذا فإن :

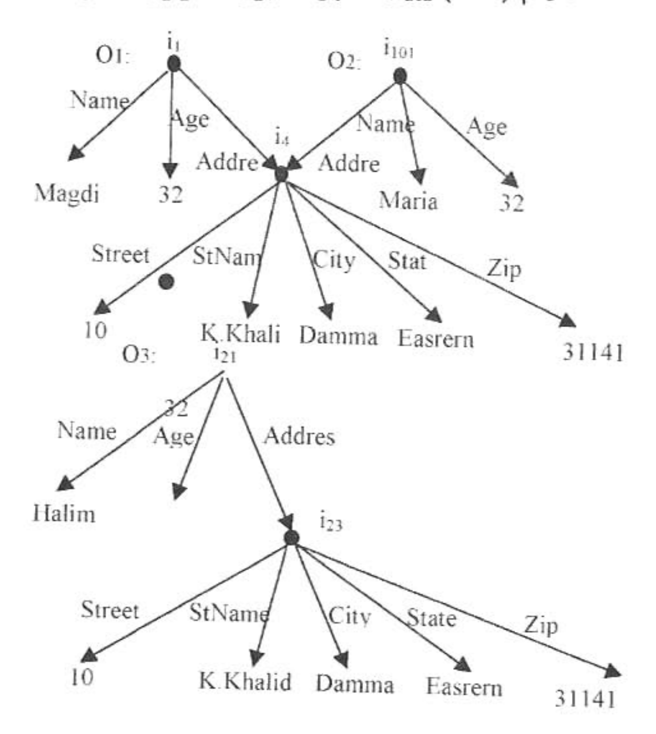
O3. ADDRESS= = O1. ADDRESS

وكذلك :

O3. ADDRESS= = O2. ADDRESS

هما شرطان غير متطابقين ، أى أن نتيجة هذا الشرط غير صحيحة.

شكل رقم (٧-٧) يبين تطابق عناوين الشيئين O1 و O2



التساوى السطحى Shallow-Equality :

ويتساوى الشيئان سطحيًا لو كانت حالتهما States أو محتوياتهما Contents متطابقتين. ويبين الشكل رقم (٧-٨) شيئين لهما أشياء مختلفة مثل قيم متغيرات الأبناء Children وفى هذه الحالة:

O1.Children = = O2. Children

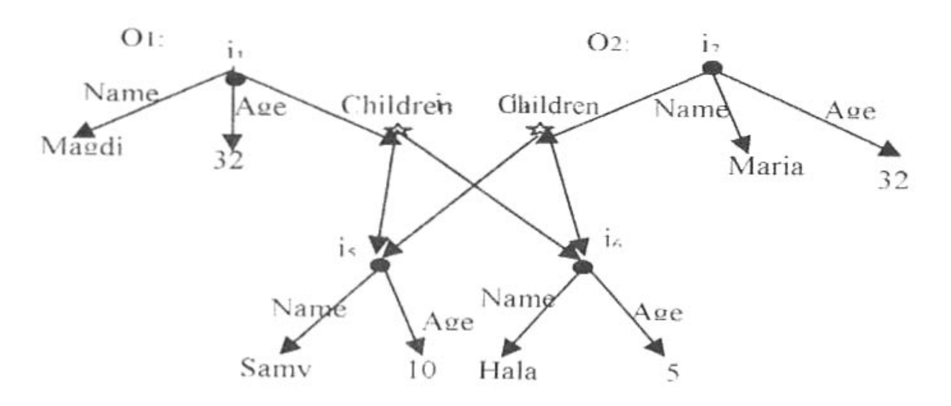
يكون هذا الشرط غير صحيح.

ومع ذلك محتويات الشيء O1 متطابق مع محتويات الشيء O2 والشيء O1 والشيء O2 لهما نفس أشياء الأبناء Children objects : ومن ثم فإن :

O1.Children == O2. Children

يكون شرطًا صحيحًا.

شكل رقم (٧-٨) يبين التساوى السطحى لأبناء الشيئين O1 و O2



التساوى العمقى Deep-Equality :

فى أبسط أشكاله يتجاهل هويات الأشياء ويتأكد مما إذا كان:

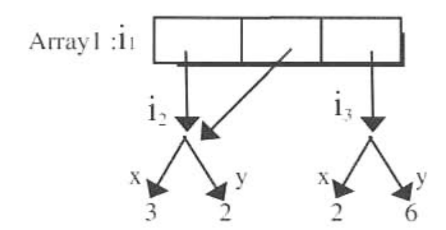
١- الشيئان وقعتين من نفس النوع Class (أى لهما نفس الهيكل أو النوع الأساسى).

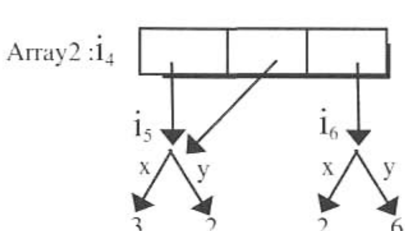
٢- قيم الأشياء الأساسية المناظرة متساوية.

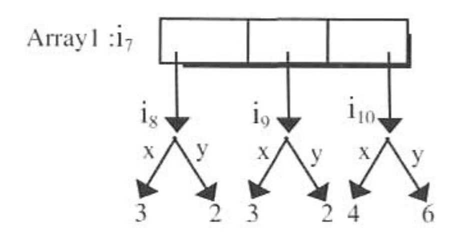
ويوضح الشكل رقم (٧-٩) ثلاث منظومات arrays لها التساوى العمقى كل للأخرى، وكل منظومة لها بعد واحد one dimensional وثلاثة عناصر. وكل عنصر لكل منظومة هو واقعة للنقطة xy. ويتبين من الشكل أن:

المنظومة الأولى Array1 بالتساوى العمقى للمنظومة الثانية Array2 يكون شرطًا صحيحًا. فى حين أن المنظومة الأولى Array1 بالتساوى العمقى للمنظومة الثالثة Array3 والمنظومة الثانية Array2 بالتساوى العمقى للمنظومة الثالثة Array3 كلاهما يكون شرطًا خاطئًا.

الشكل رقم (٧-٩) يبين منظومات التساوى العمقى







ب : العمليات الخاصة بالنسخ :

النظم الشيئية الموجهة التى تدعم هوية الشىء تعزز مظهرين لنسخ الشىء هما:

* النسخ السطحى Shallow-Copy

* النسخ العمقى Deep-Copy

وكلا العمليتين ينشىء ويرجع شيئًا جديداً new Object.

النسخ السطحى Shallow - Copy :

تنشئ رسالة النسخ السطحى شيئًا جديداً له متغيرات ذات قيم متطابقة لمتغيرات الشىء المستهدف ؛ ومن ثم فإن: O1 = O2 نسخ سطحى. ومنه يكون شرطًا صحيحًا أن: O1 تساوى سطحى O2.

النسخ العمقى Deep - Copy :

تنشئ رسالة النسخ العمقى شيئًا جديداً له متغيرات بقيم جديدة كلية حيث إن الشىء هو تساوٍ عمقى للشىء المستهدف. ويبين الشكل (٧-٩) أن المنظومة الثانية Array 2 يمكن أن ينشئ خلال رسالة النسخ العمقى: المنظومة الأولى Array2 المنظومة الثانية = Array1 نسخ عمقى.

وفى حدود التطبيق يمكن دعم الشكل الأضعف للنسخ العمقى الذى يتعرج الشىء وينشىء نسخًا لكل شىء فرعى كقيم المتغيرات أو العناصر فى مجموعة الأشياء وهكذا:

المنظومة الأولى = Array1 المنظومة الثالثة Array3 نسخ عمقى.

(ج) العمليات الخاصة بالدمج والتبديل Merging & Swapping :

بالإضافة إلى العمليات المرتبطة بدعم هوية نماذج الشىء ، فإن واحدة من أكثر العمليات أهمية هى الدمج merging. فعلى سبيل المثال: فى حالة وجود شيئين بهويتين منفصلتين، قد يكتشف أخيراً أنهما متطابقان، ومن ثم تكون هناك حاجة لدمجهما. ويلاحظ أن الدمج هو عملية تعديل لها دلالية ذات دعم قوى وصعب التطبيق. وأبسط طريقة هى أن يتم فرض المطلبين الآتيين :

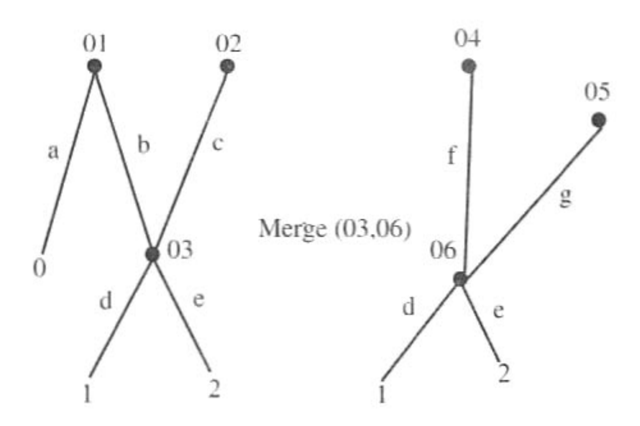
أ- الشيئان يجب أن يكونا واقعتين لنفس النوع Class ومن ثم لهما نفس النوع الأساسى Type .

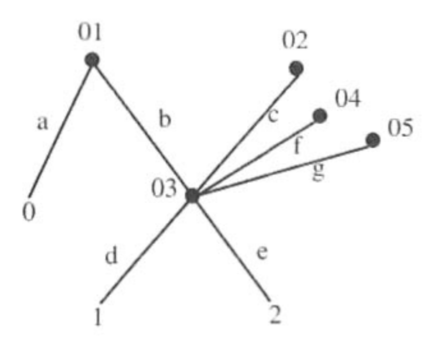
ب- الشيئان يجب أن يكون لهما التساوى العمقى .

ويبين الشكل رقم (٧-١٠) دمج مجموعات القيم المرتبة O6 و O3.

حيث إن O1.b ، O2.c قبل الدمج كان يشيران إلى نفس الشيء O3، وهكذا كان أيضاً كل من O4.f , O5.g يشيران إلى نفس الشيء O6. والشيئان O3 , O6 هما شيئان مختلفان. أما بعد عملية الدمج فإن الأشياء O1.b, O2.c , O4.f , O5.9 أصبحت تشير إلى نفس الشيء.

شكل رقم (٧-١٠) يبين دمج الشيئين O3 و O6





٥- مميزات هوية الشيء Advantages of Object Identnty :

تعرض هوية الشيء العديد من المزايا منها:

١- أنها تسمح بالتمثيل المباشر للأشياء برسم هيكل الشيء graph-Structured.

٢- لا يحتاج المستخدمون إلى الاحتفاظ بقيود السلامة .

٣- توفر مختلف العمليات المرتبطة بهوية الشيئ معالجة قوية للشيء وظائفيًا لاشياء مكاتب الذكاء.

ثانياً : إمكانيات قواعد البيانات الشيئية الموجهة DB Capabilities of object-Oriented Databases :

فيما يلى توضيح إمكانيات قواعد البيانات بشكل مختصر لتقنية قواعد البيانات الشيئة الموجهة التى سوف يتم التطرق لإمكانية الاستمرارية بشكل مسهب، حيث تنفرد قواعد البيانات الشيئة الموجهة بها عن غيرها من نظم قواعد البيانات الأخرى. ويمكن تعريف هذه الإمكانيات طبقًا للمعادلة التالية(٤):

إمكانيات قواعد البيانات = الاستمرارية + التزامنية + المعاملات + المعالجة + الاستعلام + الإصدار + الأمن +السلامة + الأداء

Database Capabilities = Persistance + Concurrency + Transaction + recovery + query + Versioning + integrity + Security + Perfarmance

الاستمرارية Persistence :

هى قدرة الأشياء على الاستمرار خلال مواضع التنفيذ المختلفة invocations للبرنامج. ومعالجة البيانات التى تتم بواسطة قاعدة البيانات الشيئية الموجهة يمكن أن تكون زائلة أو مستمرة. وتكون البيانات الزائلة (قصيرة الأمد) صحيحة فقط داخل البرنامج أو المعاملات، وهذه البيانات تزول بمجرد قطع البرنامج أو المعاملة. فى حين تخزن البيانات المستمرة خارج المعاملة وتحدث دائماً. بمعنى آخر تستمر هذه البيانات عبر المعاملات أو تحطم النظام أو حتى تحطم الوسائط.

وظيفة نظم إدارة قواعد البيانات هى أن تسمح بتداول التزامنية والتحديث لقواعد البيانات المستمرة . ولضمان استمرارية البيانات أطول فترة ممكنة، فإن نظم إدارة قواعد البيانات تطبق مختلف سياسات المعالجة قبالة المعاملات، والنظام، وتحطم الوسيط.

وتوجد علاقة ربط أساسية بين مشاركة التزامن والاستمرارية فى قواعد البيانات . تحديثات المعاملة يجب أن تستمر ، ولكن استمرارية قواعد البيانات يتم تداولها وتحديثها بشكل متزامن فإن نظم إدارة قواعد البيانات يجب أن تركز على نفسها بتناسق أشياء البيانات المستمرة [٤]،[٧].

(أ)- مستويات الاستمرارية Levels of persistence :

يمكن أن تكون معالجة البيانات فى قواعد البيانات الشيئية الموجهة زائلة transient أو مستمرة Persistent . وتكون البيانات الزائلة صحيحة فقط داخل البرنامج أو المعاملة ، وتفقد بمجرد قطع أو إنهاء البرنامج أو المعاملة. من ناحية أخرى يتم تخزين البيانات المستمرة خارج سياق " Context " البرنامج، ومن ثم تستمر فى مختلف مواضع تنفيذ البرنامج.

(ب)- سياسات الاستمرارية Persistence Strategies :

يوجد العديد من السياسات التى تشير إلى أى من الأشياء ينبغى أن يصبح مستمرة. والسياسات التالية تستخدم لإنشاء وتعريف الأشياء المستمرة :

- الامتدادات المستمرة Persistent Extensions :

يوجد افتراض أساسى فى نظم إدارة قواعد البيانات لفكرة الامتداد المستمر . ففى نظم إدارة قواعد البيانات التقليدية (مثل قواعد البيانت العلاقية)، عندما يعرف المستفيد المخطط الذى يستعمل فى لغة تعريف البيانات مثل DDL للغة الاستعلام البنائية SQL - يدمج التعريف كلاً من الهيكل والامتداد. أما فى اللغات الشيئية الموجهة، يعرف المستفيد هيكل الأشياء خلال تشييد النوع Class. حيث إن النوع يمثل مصنف الأشياء التى لها نفس النوع الأساسى type ، ويكون مقيداً فى امتدادات النوع

المستمرة وأن تعالج النوع Class كمشيد لنوع البيانات التجريدية وبوصفه إناء حاويًا لكل وقائعه. تدعم نظم قواعد البيانات الشيئية الموجهة الامتدادات المستمرة بمكررات Iterators أو تشييدات أخرى لتبحر الوقائع المستمرة للنوع.

وقواعد البيانات الشيئية الموجهة التي تدعم الامتدادات المستمرة لها على الأقل ثلاثة اساليب يتم بواسطتها إنشاء الشيء :

١- واقعة النوع المستمر في كل مرة يتم إنشاؤها ، يتم إضافتها تلقائيًا في الامتداد.

٢- يمكن أن يتم توصيف الشيء مستمرًا لحظة إنشائه . فعلى سبيل المثال: المشيد "جديد" New الخاص بالشيء.

٣- يمكن أن يطلب عند تخزين الشيء في قاعدة بيانات مستمرة صراحة من خلال عمليتي الكتابة Write والتخزين Save.

*** الاستمرارية خلال القدرة على التمديد Persistence Through Reachability :**

تدمج لغات البرمجة وقواعد البيانات مختلف مشيدات النوع للصفوف للقيم المرتبة (السجل ، record ، التجميع aggregate ...) ومجموعات الأشياء (الفئة Set، الامتداد extension ، المجموعة group ...) . ولذلك فإن القدرة على التمديد يمكن تعريفها بشكل انتقالي transitive باستعمال نماذج هذه الفئات والصفوف . قاعدة البيانات هي أصل root حيز الشيء المستمر، وكل شيء قابل للتمديد من اصل هذه القاعدة يكون مستمرًا. يوضح الشكل رقم (٧-١١) حيز الشيء الزائل والمستمر القابل للتداول في المعاملة. ويلاحظ أن الأشياء المستمرة يمكن أن يكون لها أشياء فرعية لأشياء زائلة ، لكن الشيء المستمر يمكن أن يكون له أباء متعددون multiple parents ، والبعض منهم قد تكون زائلاً. والأشياء في حيز الشيء الزائل تكون مرئية فقط داخل المعاملة الجارية . وعندما تنتهي المعاملة فإن الأشياء الزائلة لا تظهر.

شكل رقم (٧-١١) حيز الشى المستديم والزائل

Persistent Object Space	Transient Object Space
Database	X Y Z

- **الوقائع** المستمرة Persistent Instances :

وهذه السياسة تعالج وقائع خاصة للنوع، وذلك باستمراريتها أو بتوصيفها بشكل صريح على أن تكون مستمرة أو بجعلها داخل شيء مستمر وتستدعى من خلال وظيفة (دالة) Function . وعلى سبيل المثال: الفئة التى تمثل فئة وقائع موجودة لنوع مجلد Folder أثناء وقت تنفيذ البرنامج . فمطور البرنامج التطبيقى قد يوصف بعض الوقائع لكى تكون مستمرة والبعض الآخر غير مستمر. ففى التوصيف التى تكون فيه الوقائع (الأشياء) مستمرة تكون مستقلة تماماً عن تعريف النوع Class.

٢- حيز الشيء المستمر Persistent Object Spaces :

يشير مصطلح حيز الشيء المستمر إلى مجموعة كل الوقائع التى تكون مستمرة، وفى النظم الشيئية الموجهة مجموعة كل الوقائع المستمرة تكون وقائع للأنوا ع. وأحد المظاهر الأساسية للاستمرارية فى لغات البرمجة والنظم الشيئية الموجهة هو القدرة لجعل المرجعيات references أو مؤشرات الشيء Object Pointers مستمرة. والعديد من السياسات التطبيقية يمكن أن تستعمل لدعم هوية الشيء ، التى تشمل استعمال مفاتيح المعرفات واستعمال العناوين الافتراضية والمادية لهوية الشيء. وهذه السياسات يمكن أن تستعمل كتقنيات تطبيقية ضمنية تدعم بشكل أكثر كمالاً مفهوم هوية الشيء.

هناك اعتباران مهمان فى تطبيق الهوية:

أ - حيز الشىء الزائل عكس المستمر.

ب- سياسات العنونة عكس اللامباشرة.

(أ) حيز الشىء الزائل عكس المستمر:

يذكر هذا الاعتبار للاكتمال. والسبب الرئيسى لامتلاك حيز الشىء المستمر فى التطبيق الضمنى هو أن تدعم قواعد البيانات ولغات البرمجة المستمرة. والسبب الآخر الخاص بتدعيم معرفات التخزين الثانوى التى تؤدى إلى توفير التداول لأكبر حيز شيئى.

(ب) العنونة عكس اللامباشرة:

ويوفر هذا الاعتبار تصنيف أكثر قوة للتمييز بين مختلف سياسات التطبيق والعنونة يمكن ان تكون:

- عنوان الذاكرة الافتراضية.
- عنوان وسيط التخزين الثانوى.
- اسم مهيكل فى بيئة موزعة.

الهوية خلال العنونة :

أبسط تطبيق لهوية شىء ما هو أن يستعمل عنوان الشىء كهوية واستعمال العناوين الافتراضية لهوية الشىء ليس معوقًا للاستمرارية ، وذلك لسببين هما:

- التمثيل الازدواجى:

إنه من الممكن أن يحدث تمثيل هوية التطبيق أو تمثيل حالة الشىء للأشياء المستمرة لمقيمى التخزين - الثانوى وتمثيل آخر مختلف عندما يكون الشىء مخزنًا فى الذاكرة الرئيسية.

- التخزين المستمر :

إنه من الممكن أن يحدث تخزين مستمر لحيز العنوان ويتم استعماله مثل أى حيز عنوان افتراضى. وليس هناك اختلاف فى استخدام المرجعية أو التداول للأشياء التى يتم تخزينها فى حيز عنوان مستمر مقارنة بالعناوين فى الذاكرة الرئيسية.

أما اللامباشرة يمكن أن تكون:

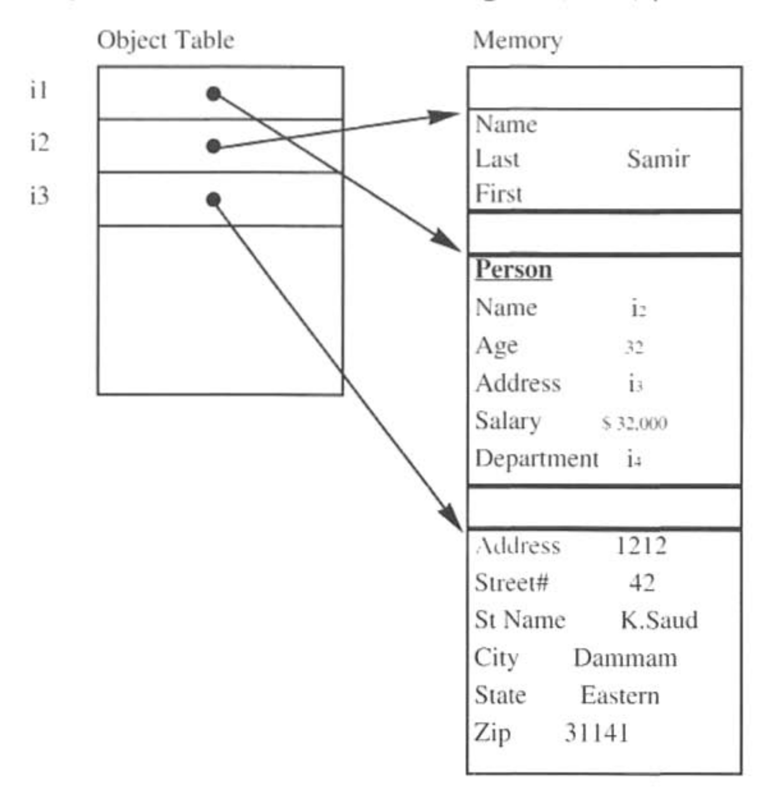
- خلال جدول مقيمى - الذاكرة memory-resident .

- خلال فهرس الأشياء الخاص بمقيمى - التخزين - الثانوى Secandary-Storage-resident.

اللامباشر خلال جدول الشىء :

واللامباشر خلال جدول الشىء الذى به معرف الشىء هو فهرس أو مؤشر لمدخل فى هذا الجدول . وهذا المدخل entry يحتوى على عنوان البداية للشىء كما هو موضح بالشكل رقم (٧-١٢).

شكل رقم (٧-١٢) يوضح المعرفات بوصفها كشافات جدول الشىء



ومع ذلك فإن هذه السياسة تتضمن مزيد من تداول الذاكرة وذات مزية فى حرية حركة الشىء دون تأثير على هويته.

مزايا الطريقة الشيئية للشىء الموجهة :

هناك العديد من المزايا للطريقة الشيئية للشىء الموجه كما يلى[٩]:

١- القدرة على التمديد Extendibility :

إن أنواع الأشياء من حيث طرقها و (برامجها) Methods يمكن تعديلها حسب الاحتياج . مثل هذه التغيرات تكون متمركزة لنوع الشىء الواضح؛ ومن ثم فهى سهلة جداً عن النظم المبنية على أساس السجل record-based، حيث إن كثيراً من أنواع السجلات قد تكون متناثرة. بالإضافة إلى أنواع الأشياء الجديدة وبرامجها يمكن دمجها فى النظام.

٢- القيود السلوكية Behavioral Constraints :

بسبب الكبسلة ، فإن سلوك نوع الشىء النهائى يتم تحديده مسبقًا بمجموعة ثابتة من البرامج . ومن ثم فإن عمليات قواعد البيانات يتم تقيدها لكى تكون داخل هذه المواصفات السلوكية.

٣- مرونة تعريف النوع Flexibility of type Definition :

يكون المستخدم غير محدد لمفاهيم النمذجة لنموذج البيانات، ولكن يمكن تعريف العديد من أنواع البيانات المختلفة ، كل منها ذات صفات لا تتكرر.

٤- قوة النمذجة Modeling Power :

تعتبر الوراثة لكل من الخصائص Attributes والبرامج Methods أداة قوية لنمذجة البيانات. وبالعموم فإن تجريد التعميم والتخصيص، والتعريف والتجميع دعم جيد لنماذج البيانات الشيئية الحالية.

عيوب الطريقة الشيئية الموجهة :

١- فقدان الارتباطات Lack of Associations :

تجديد الارتباط الذى يتم تمثيله بواسطة أنواع علاقات الربط فى نموذج كينونة – علاقة المطور EER ليس دعماً بشكل مباشر، ويتم إنجازه بشكل غير مباشر بسماح لمراجع الشيء المتداخل Interobject. وهذا هو الضعف الوراثى للطريقة للشيئية الموجهة حيث يتم معالجة كل شيء كوحدة محتوية للمعلومات بواسطة نفسها Self-Contained.

٢- صلابة السلوك Behavior Rigidity :

تدوين التحديد والتوصيف السابقين لكل العمليات بواسطة مجموعة ثابتة من البرامج هو قيد صلب وقاسٍ ، لأنه عداد يحبذ أن يستنبط تقنية قواعد البيانات الطبيعية.

٣- لغة الاستعلام ليست عالية المستوى :

لا يوجد لغات استعلام ذات مستوى عالٍ لنماذج البيانات الشيئية الموجهة الحالية. وإن قوة وأناقة لغات الحساب العلاقى والجبر العلاقى ليس لها تشابهات فى النظم الشيئية الموجهة حتى الآن.

الهوامش :

1 - [BOOCH,1991], Grady Booch, 'Object-Oriented Analysis and Design, with Applications', Redwood City, Calif.: Benjamin-Cummings Publishing, Inc., 1991.

2 - [BROWN, 1989], A.W. BROWN, 'From Semantic Data Models to Object Orientation in Design Databases', **Information and Software Technology**, Vol. 31, number 1, January/February 1989.

3 - [MCLEOD, 1987], Hammer M. Mcleod, 'Database Description with SDM: A Semantic Database Model', **ACM Trans. Database Syst.**, 19,3,Sept.1987

4 - [KHOSHAFIAN, 1993], Setrag Khoshafian, **Object-Oriented Databases**, John Wiley and Sons,Inc.,1993.

5 - [NAVATHE, 1992], Shamkant B. Navathe, 'Evolution of Data Modeling for Databases', **Communications of the ACM**, Vol.35,No.9,September 1992.

6 - [CAMPBELL,1993], Roy H. Campbell, Nayeem Islam,David Riala and Peter Madany, 'Designing and Implementing choices: an Object-Oriented System in C++',**Comm. Of the ACM**, Vol.36,No 9,September 1993,

7 - [Wu 1999], C. Thomas Wu, **An Introduction to Object-Oriented Programming** with Java, McGraw-Hill Inc. New York,1999.

8 - [GRHAM, 1991], Ian Grham, BIS Applied Systems, **Object-Oriented Models**, Addison-Wesley Publishing Company Inc., 1991.

9 - [RIAD, 1994], Mokhtar B. Riad and Saber Abd Allah, 'Object-Oriented Databases: Features, Capabilities, Products and Development Trends', The 19th International Conference for Statistics, Computer Science, Scientific and Social Applications, Cairo, 9-14 April, 1994.

10- [DATE,1995], C.J. Date, **An Introduction to Database Systems**, Volume I, Sixth Edition, Addison-Wesley publishing Company Inc.,1995.

11. [ULLMAN,1988], Ullman J.D., **Principles of Database and Knowledge-base System**, Vol. 1, Computer Science Press, 1988.

الفصل الثامن

تقنيات مستقبلية للشيء الموجه

مقدمة :

سوف نتطرق فى هذا الفصل لعدد من الموضوعات المهمة ذات الصلة بالتقنيات المستقبلية للنظم الشيئية الموجهة. ولازال بعض من هذه التقنيات تحت الدراسة والبحث، والبعض الآخر تمت بلورته فى صورة منتجات يتم استخدامها فى التطبيقات المتعلقة بالنظم الشيئية الموجهة. وسوف يتم استعراض الموضوعات التالية فى هذا الفصل:

الشيء العلاقى Object- Relational :

سبق توضيح أن نمذجة الشيء هى توصيف النظام من خلال الأشياء التى لها هوية وسلوك وحالة مغلفة (مكبسلة). وأن النموذج العلاقى يصف النظام بواسطة المعلومات. والسؤال: كيف يمكن للنموذج العلاقى أن يدعم نمذجة الشيء؟ وستتم الإجابة على هذا السؤال من خلال استعراض الجزء المتعلق بهذا الموضوع. كما سيتم استعراض النمذجة المفاهيمية للشيء العلاقى بشكل مبسط، حيث يتم استخدامها من قبل مصممى ومطورى النظم الشيئية الموجهة.

لغة الاستعلام البنائية ٣ SQL3 :

هناك عدد من المجموعات التى مازالت تعمل تجاه تعريف قاعدة البيانات الشيئية الموجهة بشكل معيارى. وذلك لكى يتم استعمالها كأساس لإنشاء وتركيب منتجات نظم إدارة وتركيب منتجات نظم إدارة قواعد البيانات الشيئية الموجهة. وسوف يتم تقييم عمل مجموعتين منها.

المجموعة الأولى:

تركز على لغة الاستعلام البنائية ٩٢ المعيارية والخاصة بمعالجة الشيء. وسوف يتم استعراض الامتدادات الأساسية للغة الاستعلام البنائية ٣ SQL3 فى هذا الفصل.

أما المجموعة الثانية:

مجموعة إدارة قاعدة البيانات الشيئية ODMG :

وتمثل هذه المجموعة اتحاد بائعى قاعدة البيانات الشيئية، وتسمى مجموعة إدارة قاعدة البيانات الشيئية ODMG. ولقد دأبت هذه المجموعة على استمرارية العمل طبقًا لمعايير نظم إدارة قواعد البيانات الشيئية ODBMSs. وقد قامت هذه المجموعة بإنتاج العديد من الإصدارات آخرها الإصدار٣. وسوف يتم استعراض المحتويات الرئيسية والتى تشمل : النموذج الشيئى، لغات توصيف الشىء ، لغات الاستعلام الشيئية وأربطة عملية تنفيذ اللغة.

لغة الأويال OPAL :

تعد الأويال OPAL لغة نظم قواعد البيانات جيم استون Gemstone وهى من أحسن نظم قواعد البيانات الشيئية الموجهة؛ لذا سوف يتم استعراض أوامر تلك اللغة بشكل مبسط.

النمذجة المفاهيمية للشىء العلاقى Object Relational :

تنشئ نمذجة الشىء نموذجًا مفاهيميًا يمكن استخدامه بواسطة مصممى النظم؛ مما يؤدى إلى إنشاء نماذج منطقية ومادية مختلفة. وسوف يتم توضيح النموذج الشيئى العلاقى ORM باستخدام المثال رقم (٨-١) الخاص بتركيب الأشياء داخل النموذج الشيئى العلاقى ORM، حيث إن "الغرفة Room" التى يتم التدريس بها، و "الفصل الدراسى Class" المرتبط بها[١].

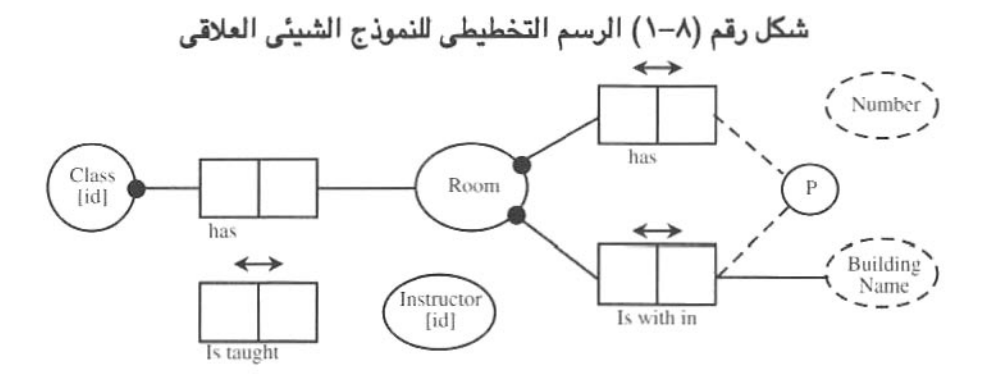
وسوف يتم تناول النموذج المنطقى فى هذه الحالة كنموذج علاقى وكنموذج شيئى علاقى. وسيتم توضيح استخدام لغة الاستعلام البنائية SQL لخيار النموذج الشيئى العلاقى باستعمال لغة أوركل ٨ Oracle 8.0. وتظل حقائق النموذج المفاهيمى بشكل لا يتغير فى كل من النموذج العلاقى والنموذج الشيئى العلاقى.

مثال (٨-١) :

يتم تعريف الشىء المركب Composite Object الذى يشمل "الغرفة" Room باستخدام "اسم المبنى" building Name و "رقم الغرفة" Room Number معًا كما هو

مبين فى النموذج المنطقى بالشكل رقم (٨-١). وعند ترجمة النموذج المنطقى ، فإن الشيء الخاص "بالغرفة" يظهر فى جدول الفصل الدراسى Class كخاصية (أى كعمود).

شكل رقم (٨-١) الرسم التخطيطى للنموذج الشيئى العلاقى



يبين الشكل رقم (٨-٢) الشيء الخاص بالغرفة Room والذى يتم مناظرته فى جدول الفصل الدراسى Class. ولا يلعب الشيء الخاص "بالغرفة" Room دوراً وظيفياً سوى أنه مرجعى.

شكل رقم (٨-٢) النموذج العلاقى للفصل الدراسى Class

Class	
PK	Class id Room Num Building Name Instructor id

وعندما يتم إعادة بناء النموذج المنطقى فى الشكل رقم (٨-٣) للحصول على النموذج الشيئى العلاقى ORM ، فإنه يمكن رؤية الشيء الخاص بالغرفة Room بوضوح.

شكل رقم (٨-٣) النموذج الشيئ العلاقى للفصل الدراسى Class

Class	
PK	Class id
	Room: Num, Building Name
	Instructor id

توصيف النموذج الشيئى العلاقى باستخدام Oracle 8.o :

```
---- Create Object type 'Room-Type'
Create type Room-Type as object (<<< Something New
" Create Type" Number Varchar 2(10)) :
---- Create new table Class
Create table class (
"Class ID" Varchar 2(10) not null, <<<<
Note the data type here
"Instructor ID" Varchar 2(10) null, Constraint Class-PK primary key ("Class ID");
```

يلاحظ أن إنشاء الشيء يسمى Room-Type، بينما فى جدول الفصل الدراسى Class يكون نوع البيانات هو Room-Type. والفكرة هنا هى إنشاء قواعد بيانات للأشياء ، ثم استخدام هذه الأشياء لتعريف الخصائص (الأعمدة).

الشيء العلاقى Object Relational :

يتضح من نمذجة الشيء الأولية أن النمذجة العلاقية ليست لديها الوسيلة لتمثيل الأشياء مباشرة. حيث إن المجموعات المرتبة ليس لها هوية ولا تغليف (كبسلة). وإن قيم خاصية أى مجموعة مرتبة تكون مكبسلة فقط كقيم، ومن ثم ليس لهذه القيم هوية ولا حالة. ويتم تسمية ذلك بشكل متكرر عدم التساوى Impedance-mismatch بين الأساليب الشيئية وقواعد البيانات العلاقية.

ولكن ليست هذه هى الحالة الحقيقية؛ لأنه من البديهى يجب أن يكون لأى نموذج بصفة عامة القدرة على توصيف نموذج العالم. ومن ثم يجب أن يكون لقواعد البيانات

العلاقية القدرة على توصيف حالة النموذج الشيئى. ولرؤية ذلك، فإنه من الضرورى أولاً تمديد حالة النموذج الشيئى إلى النموذج العلاقى بإحكام[٢].

الأشياء Objects كقيم Values :

ماذا يحدث لو سمح للأشياء أن تكون قيماً لخصائص المجموعة المرتبة Attributes of tuple. على سبيل المثال: تمثيل الأشخاص كقيم لمجموعة مرتبة يودى بسهولة إلى معرفة الشخص من خلال الاسم الأول والاسم الأخير[٢].

يتم توفير ذلك بشكل مرن، حيث إنه يمكن استعمال نوع البيانات التجريدية للحصول على شيء معين. ولكن لكى يتوافق ذلك تماماً مع النموذج العلاقى ، فإنه من الضرورى تطوير نطاق القيم Domain: لكى يكون قادراً على أخذ هذه القيم من تلك الأشياء المتوافرة بكثرة، أو أن يكون قادراً على إنشاء شيء جديد عند السؤال. وهذا يقصد به تكامل النموذجين الشيئى والعلاقى كما هو موضح فى الشكل رقم (٨-٤).

شكل رقم (٨-٤) تكامل النموذج الشيئى والعلاقى

Parent	Child
Person#1	Person#2

البيانات الزائدة عن الحد والتطبيع:

يتبين من الشكل رقم (٨-٤) تكامل كل من النموذج الشيئى والعلاقى. ولكى تبقى مشكلة: أيهما يتم سؤاله ؟ هل يتم سؤال الأب Person # 1 عن أبنائه (الطريقة الشيئية)، أم يتم السؤال عن جدول الأب الرئيسى لمعرفة أبناء الأب Person # 1 (الطريقة العلاقية). من الطبيعى، يمكن طرح السؤالين. لكن عندئذ كيف يمكن التأكد من التغييرات التى تمت لهذا أو لتلك عند حدوث حالة التزامنية؟ بشكل واضح. إضافة إلى ذلك سوف يتم الحصول على مشكلة البيانات الزائدة، أى اللاتطبيع Non-Normalization بين الأشياء بخصائصها والمجموعات المرتبة فى الجداول العلاقية.

يتم مشاهدة هذه المشكلة بشكل خاص فى أنواع كينونات جدول علاقى ممثل فى نموذج كينونة – علاقة، حيث إن كل صف يسرد خصائص الشىء. ويوضح الشكل رقم (٨–٥) تلك المشكلة.

الشكل رقم (٨–٥) يمثل خصائص الشيئ كجدول علاقى فى نموذج كينونة – علاقة (افتراضى)

Object	SSN	F_N	L_N	Employer
Person#1	123	Adel	El_Aied	Person#3
Person#2	253	Ahmed	El_Saleh	Person#9

يمثل الشكل رقم (٨–٥) نموذجًا علاقيًا؛ لذا من الضرورى أن تأخذ المعالم العلاقية الأولوية. ولكن لكى يتم تمديد النموذج العلاقى بشكل كافٍ، فمن الضرورى تمثيل الأشياء بداخلة. حيث إن النموذج العلاقى له طريقة كاملة لتغيير حالة قاعدة البيانات؛ لذا لا ينبغى إضافة أى نموذج آخر. ومن ثم لا ينبغى تغيير حالة الشىء خلال الطرق methods الخاصة بالشىء، بل يجب تعديل متغيرات الجدول العلاقى المناسبة (بإضافة – حذف – إحلال – تغيير مجموعة مرتبة)؛ لكى يتم الحصول على التغييرات الملائمة.

خصائص الشىء كمنظورات:

هل يسمح لنوع الشىء أن يجيب سؤال يتعلق بخصائصه؟ ينبغى أن يكون ذلك ملائمًا. بدلاً من البحث فى الخصائص الأساسية للجدول الرئيسى للآباء وجدول الشخص. يمكن سؤال الشخص الأول Person # 1 عن الاسم الأول والاسم الأخير. فى حالة اتباع الطريقة العلاقية فإن نوع الشىء الخاص بالشخص Person ينبغى أن يوفر رؤية مركزية على كل الجداول التى يمكن أن تشير إلى نوع الشىء الخاص بالشخص Person، أى كل الجداول التى لها خصائص بنطاق القيم والتى تحتوى على أى شخص.

بفرض أن الشخص الأول Person # 1 يتم تمثيله فى نوع الشىء الخاص بالأشخاص Person. عندئذ يمكن الاستفسار كالتالى:

```
Select Person.SSN
From person
where person.Object = < Person # 1 >
```

أى أنه يمكن أن تتم الصياغة كالآتى :

```
Select < Person # 1 >.SSN
```

السلوك:

لإضافة سلوك معين لنوع شيء، فإنه يتطلب القدرة على توصيف طريقة التنفيذ لأى شيء معطى يشبه الشيء التالى:

```
Create Domain person Class (SetName (NewName: String)
as update
where Object = this
```

من ثم يمكن توفير كل الإمكانيات لنمذجة المعلومات ونظم التخزين وإضافة السلوك بشكل سهل للأشياء، والجداول، وقواعد البيانات ككل. هذه هى الأنوا ع الثلاثة للأشياء التى يمكن أن تمتلك سلوك داخل النظام الشيئى العلاقى.

الوراثة:

أى نوع يتم بناؤه على الوراثة، يمكن استعماله كجزء من قيود السلامة: لذلك عند توصيف نطاق القيم كنوع خاص، فإنه ينبغى أن يسمح للأشياء بأن تنفذ ذلك النوع أو أى نوع فرعى لكى تكون قيمًا فى ذلك النطاق. يسمح ذلك بمرونة وسلامة تشبه تلك المتوافرة فى لغات البرمجة. على سبيل المثال: لو كان لدينا نوعان Classes. أحدهما للأشخاص Persons والآخر للموظفين Employees ، فإنه يمكن إنشاء نطاق القيم كالتالى :

```
Create Domain Person Class {----}
Create Domain Employee Class Extends Person {----}
```

عندئذ يمكن أن يكون للشيء من أى نوع Class قيمة خاصة لنوع الشيء الخاص بالأشخاص Person. وبالتالى النوع Class الذى يتم بناؤه على الوراثة، يتم استعماله لتسهيل إنشاء الأنوا ع الفرعية التى ترث الخصائص والسلوك للأنوا ع الأصلية.

ولكن ماذا عن الوراثة بين الجداول؟ الوراثة بين الجداول لا تعد وراثة على الإطلاق، إنها فقط إدارة ضغط Compression لجداول علاقية عديدة. وتكون الجداول مترابطة بواسطة الخصائص المنطقية المشتركة. وتعتبر وراثة الجدول المضغوط ليس لها تأثير على الجداول المترابطة. إنما هى فقط طريقة بسيطة لتنفيذ الجداول المترابطة قدر الإمكان. ولهذا المعنى تكون أقرب إلى وراثة النوع ولكن ليس هناك من دا ع لاستخدام مصطلح الوراثة.

لغة الاستعلام البنائية ٣ SQL3 :

تعد هذه اللغة امتداداً للغة الاستعلام البنائية المعيارية٩٢ SQL92 ، والتى تتضمن دعماً لإدارة قاعدة البيانات الشيئية الموجهة. وتعتبر لغة الاستعلام البنائية٣ SQL3 معياراً للمنتجات، وليس منتجًا بذاته. ولا توجد حتى الآن منتجات نظم إدارة قواعد بيانات تجارية تطبق هذا المعيار[٣].

ترفع لغة الإستعلام البنائية ٣ SQL3 من إدارة قاعدة البيانات التقليدية وليس من تقليدية التفكير الشيئى. واستهدفت هذه المجموعة من العمل بلغة الاستعلام البنائية٣ SQL3 توصيف معيار تجاه التوافق مع لغة الاستعلام البنائية ٩٢ SQL92. وهذا يعنى أن معالم وظائف لغة الاستعلام البنائية ٩٢ SQL92 ينبغى تضمينها مع لغة الاستعلام البنائية ٣ SQL3 . وبناء على ذلك، فإن لغة الاستعلام البنائية ٣ تتضمن تشابهات وتسهيلات قاعدة البنائية العلاقية مع معالم الشيئ المضاف إليها، كاتجاه مقابل لتسهيل قاعدة البينات الشيئة الموجهة. وتتضمن لغة الاستعلام البنائية٣ SQL3 الثلاثة مجموعات الجديدة التالية:

- دعم نوعية البيانات التجريدية.
- تطوير تعريفات الجداول العلاقية.
- عمل امتدادات للغة الاستعلام البنائية SQL لجعل لغة الاستعلام البنائية ٣ لها القدرة على إتمام جميع العمليات الحسابية.

وسوف يتم التطرق إلى هذه الموضوعات الثلاثة لفتح المجال أمام دارسي لغة الاستعلام البنائية ٣ :

أولاً : نوعية البيانات التجريدية ADTs :

كما سبق وطرح في الفصل السابق، فإن نوعية البيانات التجريدية هي بناء أو هيكل يتم تعريفة بواسطة المستخدم. ويكافئ هذا التعريف في لغات البرمجة الشيئية نوع الشيء Class.

يمكن أن تستعمل نوعية البيانات التجريدية في تعبيرات لغة الاستعلام البنائية SQL، أو تخزينها في جداول أو الاثنين معاً. وفي حالة عدم تخزين نوعية البيانات التجريدية في جدول، فإنها تكون مؤقتة حتى لو تم استخدامها في تعبير أو أكثر للغة الاستعلام البنائية.

تعرف لغة الاستعلام البنائية ٣ SQL3 نوعين من البيانات التجريدية هما: الشيء الخاص بنوعية البيانات التجريدية، والآخر هو القيمة الخاصة بنوعية البيانات التجريدية. ويكون الشيء قابلاً للتعريف وله هيكل بيانات مستقل ويخصص له معرف OID. هذا المعرف ذو قيمة واحدة لا تتغير، وتستمر طوال فترة حياة الشيء. وتعد قيم المعرف الخاص بالشيء OID مؤشرات لأشياء أخرى، ويتم تخزين قيمة المعرف OID في جدول يوفر مؤشرات لشيء. ويكون هذا مناسباً ولكنه ينشئ مشكلة عند تدمير نوعية البيانات التجريدية الخاصة بالشيء؛ لأن المعرف في هذه الحالة لن يكون صحيحاً. ولا تشير لغة الاستعلام البنائية ٣ إلى ما سيحدث في هذه الحالة بوضوح؛ لذلك يتم كتابة البرامج لكي تختبر صحة المعرف OID قبل استعماله.

أما بخصوص القيمة الخاصة بنوعية البيانات التجريدية، لا يخصص لها معرف، ولا تستطيع القيمة الوجود إلا في سياق إنشاء الشيء الخاصة بنوعية البيانات التجريدية. أما في حالة إنشاء القيمة الخاصة بنوعية البيانات التجريدية كعمود في جدول، فإنه يتم تخزينها في ذلك الجدول، ولا تشير إلى نوعية البيانات التجريدية إلا من خلال اسم ذلك الجدول. أما في حالة إنشاء القيمة في برنامج فرعي (دالة وظيفية)، فإنها تكون مؤقتة ويتم تدميرها عند ترك البرنامج الفرعي لذاكرة الحاسب. يبين الشكل

رقم (٨-١٠) مثالاً لتعريف النوع Class (الشيء الخاص بنوعية البيانات التجريدية) فى لغة الاستعلام البنائية SQL3. وسوف لا يتم التركيز فى هذا المثال على أسلوب استخدام الأوامر ، بل على مشاهدة نوعية البيانات التجريدية ADT الخاصة بالموظف والطرق methods التى سيتم تنفيذها على هذه البيانات.

```
CREATE OBJECT TYPE employee WITH OLD VISIBLE
(name VARCHAR NOT NULL.
name CHAR(7)
salary UPDATABLE VIRTUAL GET with get-salary SET WITH
change-salary.
PRIVATE
hiredate DATE
currentsalary CURRENCY
PUBLIC
ACTOR FUNCTION get-salary (:E employee) RETURNS
CURRENCY
{code to perform security processing
and return value of currentsalary if appropriate}
RETURN salary
END FUNCTION.

ACTOR FUNCTION change-salary (: E employee) RETURNS
employee
{code to perform security processing
and computer and set new currentsalary. if appropriate}
RETURN :E
END FUNCTION.

DESTRUTOR FUNCTION remove-employee (:E employee)

RETURNS NULL
{code to get ready to delete employee data}
DESTOROY :E
RETURN :E
END FUNTION.
```

الشكل رقم (٨-١٠) يوضح مثالاً لتعريف النوع Class في لغة الاستعلام البنائية ٣ SQL3 يتضمن هذا المثال البيانات الخاصة بالموظف مثل الاسم name والرقم number ، وتاريخ التعيين hiredate. ويتم تعريف هذه البيانات كما في لغة الاستعلام البنائية SQL المثبتة للأنواع الأساسية للبيانات. ويتم أيضًا تعريف البيانات الافتراضية (الطرق Methods) والتي ينبغي أن تحتوي على واحدة على الأقل لكي تتضمن نتيجة العملية الحسابية التي سيتم إجراؤها على المرتب الأساسي. وهذه الطرق هي: الحصول على المرتب الأساسي get-Salary والتغييرات التي تطرأ على المرتب Change-Salary، وإنهاء عمل الموظف Remove-Employee. في الشكل رقم (٨-١١) والخاص بجدول الإدارة Dept يتم تعريف اسم الإدارة DeptName بحد أقصى عشرة حروف Char (10) إلى جانب الأنواع الخاصة بنوعية البيانات التجريدية. وهكذا يتم استعمال نوعية البيانات التجريدية ADT كأي نوع بيانات أخر يتم تعريفه.

شكل رقم (٨-١١) يوضح تعريف الموظف Employee كنوعية البيانات التجريدية (نوع) ADT

Create table DEPT	
(DeptName	Char (10)
Manager	Employee
Admin	Employee (Instanse))

عندما يتم استعمال عمود (حقل) كنوع بيانات تجريدية. ADT ، فإنه يتم إلحاق كلمة Instance كمؤشر للنوع الشيئي لكي يتم تخزينه؛ لذلك إذا تم حذف النوع الشيئي فإنها تظل مؤشراً له.

ويتضح من الشكل رقم (٨-١١) أن عمود المدير manager لا يتم توصيفه بكلمة In-stance، بينما عمود الإداريين Admin تم توصيفه بكلمة Instance. وهذا يعني أن كل صف بجدول الإدارة Dept سوف يحتوي على مؤشر للموظف Employee في عمود المدير manager والبيانات الفعلية وطرق methods لأي موظف في عمود الإداريين Admin. ويتم استعمال خانات البيانات العامة للنوع الشيئي في جمل لغة الاستعلام البنائية ٣ SQL3 تمامًا كما تستعمل في أعمدة الجدول. ويتم توضيح ذلك من خلال صياغة جملة لغة الاستعمال البنائية ٣ التالية:

```
SELECT   DEPTNAME . MANAGER.OID ,
         MANAGER.NAME. ADMIN.OID .
         ADMIN.NAME
FROM     DEPT
```

وسوف يتم استخلاص DEPTNAME, NAMAGER.OID .ADMIN-NAME, ADMIN.OID من الجدول، وهذا يعنى أن نظام إدارة قاعدة البيانات DBMS سوف يستعمل قيمة MANAGER.OID للحصول على قيمة المؤشر Instance الخاصة بالموظف Employee. وينتزع نظام إدارة قاعدة البيانات DBMS اسم المدير Manager-Name من النوع الشيئى استجابة لجملة لغة الاستعلام البيانية٣ وهى نفس النتيجة كما لو أن نوع الشيء الخاص بالمدير Manager قد تم تخزينه فى الجدول. ولكن ينبغى توضيح أن المعرف OID الذى تم تخزينه فى الجدول قد يصبح غير صحيح لو أن نوع الشيء المرتبط به قد تم حذفه؛ ومن ثم يحتاج نظام إدارة قاعدة البيانات DBMS لعمل معالجة لهذا الخطأ. ويتضح ذلك من خلال عبارة لغة الاستعلام البنائية٣ التالية:

```
SELECT     DEPTNAME. MANAGER.NAME.
MANAGER. SALARY
 FROM  DEPT
```

ولمعالجة هذه الجملة، فإن نظام إدارة قاعدة البيانات DBMS يحتاج لتداول جدول الإدارة DEPT وعند حصوله على المعرف OID الخاص بالمدير Manager ، فإنه يحصل على قيمة المؤشر Instance الخاص بالمدير manager . عندئذ يتم تنفيذ الطريقة Method الخاصة للحصول على المرتب الأساسى get-salary ، ثم يتم ترجيع قيمة أو ترجيع خطأ لو لم توجد قيمة. وسوف يتكرر ذلك مع كل سطر فى جدول الإدارة DEPT فى الشكل رقم (٨-١١). وترتبط خانات البيانات الخاصة بنوع الشيء بالطرق الخاصة به: لذا تكون جملة لغة الاستعلام البنائية ٣ التالية غير صحيحة:

```
SELECT     DEPTNAME. MANAGER.CURRENTSALARY
FROM   DEPT
```

والأسلوب الوحيد للحصول على بيانات المرتب الحالى Carrentsalary من نوع الشيء الخاص بالموظف Employee هى من خلال طريقة get-Salary. ويمكن تخصيص قيماً للأعمدة بجمل أخرى شبيهه بلغة الاستعلام البنائية٣، كما فى التعبير التالى:

```
UPDATE   DEPT
SET    ADMIN-NAME = 'ABDEL R. EL-ARFAJ'
WHERE    DEPTNAME = 'LIBRARY'
```

ويتم تمثيل بعض الأشياء بواسطة المؤشرات وليست بواسطة قيمة البيانات. ويلاحظ أن جملة لغة الاستعلام البنائية٣ التالية لا تغير تخصيص الموظف:

```
UPDATE DEPT
SET    MANAGER.NAME = 'ABDEL  R. EL-ARFAJ'
WHERE    DEPTNAME ='LIBRARY'
```

هذه الجملة لا تغير تخصيص الموظف حيث إن الموظف الذى له اسم Abdel R. El-Arfaj قد تم تخصيصه لإدارة المكتبة فبدلاً من ذلك فإنه يتغير اسم الموظف الذى يكون حاليًا المدير. وهذا يعنى أن أى جدول أخر يشير إلى الشيء الذى يخص هذا الموظف سوف يكون له الاسم الذى يتم تغييره. لكى يتم إحلال المدير فى إدارة المكتبة بموظف آخر له اسم Abdel R. El-Arfaj ، فإن الشيء الخاص بالمدير يحتاج إلى وضعه فى قيمة الشيء الصحيحة. وسوف تتمم الجملة التالية هذا الحدث:

```
UPDATE   Dept
SET   Manager =
               SELECT Employee.OID
               FROM  Employee
               WHERE Name = 'Abdel R. El-Arfaj'
WHERE    Deptname = 'Library'
```

وبناء على ذلك فإنه هذه الجملة تعتبر صحيحة.

بناء على ذلك فإن تدعيم لغة الاستعلام البنائية٣ بنوعية البيانات التجريدية، يمكنها من القدرة على تعريف وتخزين ومعالجة أنوا ع الأشياء. وهناك تغييران فى لغة الاستعلام البنائية تم أخذهما فى اعتبارات لغة الاستعلام البنائية٣. وسوف يتم توضيحها عند استعراض امتدادات اللغة.

ثانياً - تطوير تعريفات الجداول العلاقية :

قد تم تمديد تعريف الجداول العلاقية فى لغة الاستعلام البنائية ٣ فى عدة امتدادات منها:

١- الامتداد الأول :

يتضمن هذا الامتداد ضرورة احتواء جداول لغة الاستعلام البنائية٣ لمعرف صف Row Identifier. وهذا المعرف لا يتكرر مع أى صف بالجدول ويعمل عمل المفتاح النائب Surrogate والذى تمت الإشارة إليه فى الفصل السابق. ويمكن للتطبيقات استخدام هذا المعرف بشكل صريح لو تم تضمينه فى التعبير مع الهوية Identity عند تعريف الجدول: لذا فإن أى جدول يتم تعريفه بهذا الأسلوب يجب أن يعطى عموداً ضمنيًا يسمى عمود الهوية. ويمكن استخدام قيم هذا العمود بواسطة التطبيق، ولكن هذه القيم لا يتم تضمينها فى نتائج تعبير عبارة الاختبار Select.

٢- الامتداد الثانى:

يتضمن هذا التمديد مفهوم الجدول فى لغة الاستعلام البنائية ٣. حيث يتضمن هذا المفهوم تعريف ثلاثة أنوا ع للجداول هى: جدول الفئة Set ، الجدول متعدد الفئات Mul-tisets ، وجدول القائمة List .

*** جدول الفئة SET :**

جدول لا تتكرر فيه الصفوف.

*** جدول متعدد الفئات:**

جدول قد تتكرر فيه الصفوف. ويعتبر هذا الجدول مكافئًا لمفهوم الجدول فى لغة الاستعلام البنائية ٩٢ SQL92. وفيه يتم تجاهل عمود (حقل) الهوية Identity ، حيث إنه لو تم استخدام عمود الهوية ، فإن الجدول يصير بلا تكرار للصفوف.

*** جدول القائمة List :**

جدول يتم تعريفه بواسطة عمود أو أكثر، وله ترتيب خاص.

٢- الامتداد الثالث:

يتضمن هذا التمديد مفهوم الجدول فى لغة الاستعلام البنائية ٣ SQL3. ويعد الجدول حسب هذا المفهوم جدولاً فرعياً Subtable حيث إنه يشكل فئة جزئية من جدول آخر يسمى الجدول الأصلى Supertable. ويرث الجدول الفرعى كل أعمدة الجدول الأصلى، بجانب امتلاكه لأعمدة خاصة. ويتضمن أى جدول أصلى معرف صف يتم تعريفة ضمنيًا. ففى الشكل رقم (٨-١٢) يتم تعريف نوعين للأستاذ:

Tenured-professor ، Nontenured-Professor. ويمتلك كل منهما عموداً يخص الهوية على الرغم من عدم توصيفه، ويرجع السبب وراء ذلك إلى أن كلاً منهما نوع فرعى:

شكل رقم (٨-١٢) يوضح تعريف الجدول الفرعى

```
CREATE TABLE PROFESSOR WITH IDENTITY
        (ProfessorName)      Char (10)
         Phone               Char (7)
         Office              Char (5)

CREATE TABLE TENURED-PROFESSOR  UNDER PROFESSOR
        (Date TenureGranted      Date)
CREATE TABLE NON-TENURED-PROFESSOR UNDER PROFESSOR
        (NextReveiw              Date)
```

ثالثاً : عمل امتدادات لغة الاستعلام البنائية SQL :

بناء على امتدادات لغة الاستعلام البنائية٣ SQL3، فإن الطرق Methods التى يتم تطبيقها لنوعية البيانات التجريدية، يمكن استخدامها فى تشفير لغة الاستعلام البنائية SQL. ولجعل هذه الإمكانية أكثر قوة، فإن عناصر اللغة التى يتم اقتراحها سوف تجعل لغة الاستعلام البنائية كاملة حسابيًا. ويبين الشكل رقم (٨-١٣) تلخيصاً للإضافات التى يمكن اقتراحها.

شكل رقم (٨-١٣) يبين امتدادات اللغة المقترحة

Statement	Purpose
DESTROY	Destroy an object ADT:Valid only in DESTRUCTOR functions
ASSIGNMENT	Allow the result of an SQL value expression to be assigned to a local var iable,column,or ADT attribute
CALL	Invoke an SQL orocedure
RETURN	Return a value from a value computation in a procedure or function
CASE	Select execution path on the basis of alternative values
IF THEN SLSE	Allow conditional logic
WHILE LOOP	Allow iterative logic

تعد لغة الاستعلام البنائية SQL لغة الفئات الموجهة Set-Oriented. وتتميز تلك اللغة بجملة الاختيار SELECT التى يتم استخدامها فى اختيار مجموعة من الصفوف. ويتم تغيير سمات هذه اللغة بإضافة الجمل فى شكل رقم (٨-١٣). سوف يجعل هذا التغيير لغة الاستعلام البنائية أكثر تشابهًا بلغات البرمجة التقليدية. ويعد هذا ضروريًا فى حالة استخدام لغة الاستعلام البنائية كلغة للمنطقيات المستخدمة فى الطرق Methods الخاصة بنوعية البيانات التجريدية ADT، وإن كان هذا سيؤدى إلى تغيير فى الصفات الوظيفية للغة الاستعلام البنائية SQL.

مجموعة إدارة قاعدة البيانات الشيئية Object Database Management Group (ODMG) :

يمثل مستقبل هذه المجموعة ODMG جذبًا حقيقيًا؛ نظرًا لتمكنها من إدارة عملية التخزين الدائم للأشياء بغض النظر عن آلية التخزين الفعلية، سواء قاعدة بيانات شيئية أو علاقية. إلى جانب القدرة على بناء تطبيقات الشبكة العنكبوتية Web أو الوسائط المتعددة [٤]،[٥].

أهمية المعيار:

واجهت مجموعة إدارة قاعدة البيانات الشيئية ODMG العديد من الصعوبات قبل بدء عملها. ويرجع السبب وراء ذلك إلى افتقار قواعد البيانات الشيئية إلى المعيار؛ مما أدى إلى محدودية استعمالها على نطاق واسع. فى حين يعد نجاح نظم قواعد البيانات العلاقية ليس فقط نتيجة استعمالها لنموذج مبسط عن النظم السابقة أو لاستقلالية البيانات، بل نتيجة لارتكازها على العديد من المعايير. فقد سمح قبول معيار لغة الاستعلام البنائية SQL بدرجة عالية فى إمكانية نقل وإجراء العمليات بين النظم، إلى جانب تبسيط تعلم نظم قواعد البيانات العلاقية: مما شكل مصادقة واسعة النطاق للطريقة العلاقية: لذا يعد المعيار هو الشرط الأساسى لصناعة مثل هذه التطبيقات العملية.

ولقد أعطى المجهود المكثف الذى تم بذله من قبل مجموعة إدارة قاعدة البيانات الشيئية ODMG لصناعة قواعد البيانات الشيئية "بداية القفز" نحو المعايير التى ينبغى أن تتبع، خلافاً لما حدث فى السنوات السابقة.

الأهداف:

الهدف الأساسى لمجموعة إدارة قاعدة البيانات الشيئية ODMG هو وضع مجموعة من المعايير التى تسمح لمستخدمى نظم إدارة قواعد البيانات الشيئية ODBMSs أن يكتبوا التطبيقات المحمولة، أى التطبيقات التى يمكن أن تعمل على أكثر من منتج لنظم قواعد البيانات الشيئية مثل: مخطط البيانات ، رباط عملية تنفيذ لغة البرمجة Pro-binding gramming Language، ومعالجة البيانات، ولغات الاستعلام.

وتم استنتاج عمل مجموعة إدارة قاعدة البيانات الشيئية ODMG بتجميع المعالم الرئيسية القوية لمنتجات نظم إدارة قواعد البيانات الشيئية المتوافرة حالياً بالأسواق. حيث تعرض هذه المنتجات تطبيقات تم برهنتها لمحتويات المعايير التى تم إجراء محاولات لها فى هذا الحقل.

التعريف:

قامت مجموعة إدارة قواعد البيانات الشيئية ODMG بتعريف نظم إدارة قواعد البيانات الشيئية الموجهة كنظم قواعد بيانات تتكامل فيها إمكانيات قواعد البيانات مع إمكانيات لغة البرمجة الشيئية الموجهة. وتجعل نظم إدارة قواعد البيانات الشيئية الموجهة لأشياء قواعد تبدو كأنها أشياء لغة برمجة فى أكثر من لغة برمجة متوافرة حاليًا بالأسواق. وتمتد نظم قواعد البيانات الشيئية الموجهة إلى اللغة مع البيانات الدائمة، مراقبة التزامنيية، معالجة البيانات، الاستفسارات المترابطة وإمكانيات قواعد البيانات الأخرى.

المحتويات الرئيسية:

١- النموذج الشيئى Object Model :

لقد استعملت هذه المجموعة النموذج الشيئى الذى أطلق عليه OMG والذى تم تدعيمه بواسطة نظم إدارة قواعد البيانات الشيئية. وقد تم تصميم هذا النموذج لكى يكون شائع الاستعمال لوسطاء مستخدمى الشىء، ونظم قواعد البيانات الشيئية، ولغات البرمجة الشيئية والتطبيقات الأخرى.

يصف هذا النموذج المقصود بالأشياء، والثوابت، والأنوا ع الآساسية، والعمليات ، والخصائص، وعلاقات الربط، بالإضافة إلى توصيف الأنوا ع الخاصة مثل الوثيقة، والمؤلف، والناشر، والفصل، والأنوا ع، وخصائص كل هذه الأنوا ع. ويمثل هذا النموذج مخططًا لقاعدة البيانات.

يتم التناظر بين النموذج الشيئى لمجموعة إدارة قاعدة البيانات الشيئية ODMG لقواعد البيانات الشيئية والنموذج العلاقى لقواعد البيانات العلاقية كما هو مجسد فى لغة الاستعلام البنائية SQL. ويعد النموذج العلاقى التعريف الأساسى لنظم إدارة قواعد البيانات العلاقية وظيفيًا. كذلك يعتبر النموذج الشيئى لمجموعة إدارة قاعدة البيانات الشيئية ODMG التعريف الأساسى لنظم إدارة قواعد البيانات الشيئية وظيفيًا. يتضمن هذا النموذج دلاليات أكثر من تلك المتوافرة فى النموذج العلاقى بتوصيفه لعلاقات الربط والعمليات بشكل أكثر وضوحًا.

٢- لغات توصيف الشيء Object Specification Languages :

تستخدم مجموعة إدارة قاعدة البيانات الشيئية ODMG التابعة لنظم إدارة قواعد البيانات الشيئية لغة "تعريف الشيء" ODL ولغة "شكل تبادل الشيء" OIF لتوصيف الشيء. وسوف يتم استعراض لغة تعريف الشيء والتى يتم استخدامها لتعريف التوصيفات للأنواع Classes التى تتطابق مع النموذج الشيئى لمجموعة إدارة قاعدة البيانات الشيئية ODMG.

ويتم استخدام هذه اللغة لدعم إمكانية نقل مخططات قواعد البيانات خلال مطابقتها لنظم إدارة قواعد البيانات الشيئية. وهناك العديد من المبادئ التى أرشدت إلى تطوير لغة تعريف الشيء ODL منها:

- تدعيم هذه اللغة للتشييدات المنطقية للنموذج الشيئى لمجموعة إدارة قاعدة البيانات الشيئية ODMG.
- لا ينبغى أن تكون هذه اللغة لغة برمجة شاملة، بل لغة لتعريف توصيفات الشيء.
- ينبغى أن تكون هذه اللغة مستقلة.
- ينبغى أن تتوافق هذه اللغة مع لغة تعريف واجهة التطبيق IDL.
- ينبغى أن تكون هذه اللغة قابلة للتمديد.
- ينبغى أن تكون هذه اللغة ذات قيمة لمطورى التطبيقات.

بالإضافة إلى ذلك ، توفر نظم إدارة قواعد البيانات الشيئية تسهيلات لتدعيم تعريف البيانات باستعمال لغة تعريف البيانات DDL ولغة معالجة البيانات DML. حيث يتم استخدام لغة تعريف البيانات DDL لتعريف أنواع البيانات وواجهات التطبيق. فى حين يتم استخدام لغة معالجة البيانات DML لإنشاء وحذف تعديل وقراءة قيم هذه الأنواع من البيانات.

وتعد لغة تعريف الشيء ODL هى نفسها لغة تعريف البيانات DDL للأنواع Classes. ومع ذلك لم توفر مجموعة إدارة قاعدة البيانات الشيئية ODMG لغة لتوصيف معالجة

الشيء OML. كذلك لم ترتبط لغة تعريف الشيء ODL بالتركيب Syntax الخاص للغة برمجة معينة، بل بتعريف مخطط دلاليات في لغة برمجة مستقلة.

وتم تصميم أربطة bindings لغة تعريف الشيء لعدد من لغات البرمجية هي: C++ ، Smalltalk ، Java لكي توافق توصيف تركيب لغة البرمجة المضيفة.

٣- لغة الاستعلام الشيئية OQL :

تعد لغة الاستعلام الشيئية OQL لمجموعة إدارة قاعدة البيانات ODMG لغة توصيفية وليست إجرائية. ويتم استخدام هذه اللغة للاستفسار وتحديث أشياء قواعد البيانات التي تدعم النموذج الشيئي لمجموعة إدارة قاعدة البيانات الشيئية ODMG بالإضافة إلى تعامل هذه اللغة مع الأشياء المعقدة، وقد تم بناء هذه اللغة على الفرضيات التالية:

- أن تعتمد على النموذج الشيئي لمجموعة إدارة قاعدة البيانات الشيئية ODMG.
- أن تقترب من لغة الاستعلام البنائية ٩٢ SQL92. حيث يتم تركيز الامتدادات على تدوينات الشيء الموجهة مثل:

الأشياء المعقدة، هوية الشيء، مسار التعبيرات، تنفيذ العمليات، أربطة التنفيذ المتأخرة.

- أن توفر تعليمات عالية المستوى للتعامل مع مجموعة الأشياء.
- أن تكون لغة وظيفية، بحيث يمكن للعمليات أن تكون مركبة بشكل حر.
- أن تكون لغة استعلام بسيطة الاستخدام، وتوفر التداول لنظم إدارة قواعد البيانات الشيئية؛ ولذا فهي غير مكتملة ولا تستخدم لإجراء العمليات الحسابية.
- أن يتم إدارتها بواسطة الطرق Methods المعرفة للأشياء؛ لذا فهي لا توفر عمليات تحديث صريحة.
- أن توفر تداول الأشياء، لذلك تكون الاستفسارات مثالية بشكل سهل بواسطة قوة طبيعية التوصيفات.
- أن يتم تعريف الدلاليات الشكلية للغة بشكل سهل.

٤- أربطة عملية تنفيذ اللغة Language bindings :

تعد أربطة عمليات تنفيذ اللغات التالية:

ODMG Java ، ODMG C++ ، ODMG Smalltalk امتدادات لمعايير للغات ، java ، C++، Smalltalk، على التوالى لكى تسمح بتخزين الأشياء الدائمة. ويتضمن كل رباط دعمًا للغة الاستعلام الشيئية OQL، التبحرات، المعاملات transactions. وتؤدى هذه الطريقة إلى تمكين مطورى التطبيقات من بناء تطبيق لقاعدة بيانات كاملة من بيئة لغة البرمجة الفردية.

- رباط عملية تنفيذ لغة **Java** :

يعتبر رباط عملية تنفيذ لغة Java لمجموعة إدارة قاعدة البيانات الشيئية طبيعيًا وتكميليًا لمبرمجى لغة Java. وتكون الاستمرارية بواسطة قابلية الوصول. وهذا يعنى أن أى شىء يصل عن طريق أصل root الأشياء فى قواعد البيانات، يصبح تلقائيًا شيئًا دائمًا فى قاعدة البيانات. وتضيف أربطة عملية التنفيذ للغة Java لمجموعة إدارة قاعدة البيانات الشيئية ODMG الأنواع Classes والتشييدات الأخرى للغة Java لكى تدعم النموذج الشيئى، والذى يتضمن التجميعات ، والتعاملات، وقواعد البيانات دون تغيير دلاليات اللغة.

- أربطة عملية تنفيذ لغة C ++ :

توفر أربطة عملية تنفيذ لغة C++ لمجموعة إدارة قاعدة البيانات الشيئية امتدادات واضحة لتدعيم إنشاء الشيئ والتسمية والمعالجة والحذف وإمكانية التعامل وعمليات قواعد البيانات. كما تسمح للأنواع القادرة على الاستمرارية أن تنشئ عن طريق الوراثة.

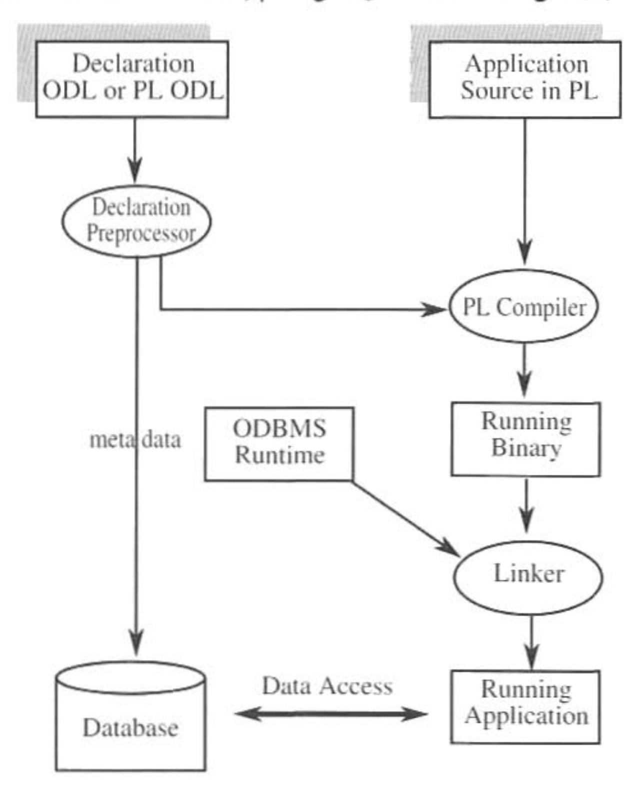
- أربطة عملية تنفيذ لغة Smalltalk :

تستطيع أربطة عملية تنفيذ لغة Smalltalk لمجموعة إدارة قاعدة البيانات الشيئية ODMG من تخزين أشياء لغة Smalltalk ، والتى يتم تصنيعها بواسطة الشىء القابل للوصول بشكل دائم عندما تتم الإشارة إليها بواسطة شىء مستمر فى قاعدة البيانات، أما الأشياء التى لم تعد ذات قيمة فانه لا يتم الوصول إليها.

الرسم المنظورى للبناء المعمارى لنظم قواعد البيانات الشيئية:

يتم كتابة توصيفات مخطط التطبيق (بيانات أو واجهة التطبيق)، بالإضافة إلى البرنامج الأساسى لتنفيذ التطبيق والذى يتم كتابته بلغة برمجة PL معينة مثل لغة ++C. فى حين تكتب توصيفات المخطط فى امتداد وتركيب لغة برمجة PL ODL أو فى لغة برمجة مستقلة ODL. ويتم ترجمة التوصيفات والبرنامج الرئيسى وإلحاقها بنظم إدارة قاعدة البيانات الشيئية الموجهة لكى يتم تنفيذ التطبيق. ويبين الشكل رقم (٨-١٤) رسمًا مبسطًا للبناء المعمارى لنظم قواعد البيانات الشيئية، والذى يوضح الاستعمال المثالى لمنتج نظام إدارة قاعدة البيانات الشيئية ODBMS الذى تحاول تلك المجموعة أن تجعله معياريًا.

شكل رقم (٨-١٤) يوضح الاستعمال المثالى لمنتج نظام إدارة قاعدة البيانات الشيئية ODBMS



لغة الأوبال OPAL :

فى نظم قواعد البيانات الشيئية الموجهة فإن معالجة البيانات Data manipulation يتم إنجازها بواسطة العمليات التى يتم تعريفها فى واجهة تطبيق النوع Class inter-face وخلال التشييدات المتوافرة للغة البرمجة المحيطة بتعريفات النوع Class. ومع ذلك فالعديد من نظم الشيئية الموجهة أيضًا توفر واجهات تطبيق للغة استعلام عالية المستوى، ومعظمها يتم بناؤه على لغة الاستعلام الهيكلية SQL . وتعد الأوبال OPAL لغة نظم قواعد البيانات جيم استون Gemstone وهى من أحسن نظم قواعد البيانات الشيئية الموجهة. وتم تسجيل نظم قواعد البيانات جيم استون Gemstone بواسطة مجموعة سرفيو لوجيك Servio Logic Corp. وتأثرت لغة الأوبال OPAL كثيرًا بلغة Smalltalk-80 فى تسهيلاتها لتعريف البيانات ومعالجتها. وسوف يتضح من أشكال لغة الأوبال OPAL أنها اكثر أهمية فى معالجة البيانات[٧]،[٨].

تعريف البيانات Data Definition :

الأنـواع Classes :

يقصد بها أنواع البيانات التجريدية ADTs وتتكون من:

١- تعريف هيكل البيانات للنوع.

٢- مجموعة من العمليات تسمى طرق (أو برامج) "Methods" والتى يتم تطبيقها على الأشياء فى النوع.

مثال ذلك: لو افترض وجود النوع الأصلى C وهناك حاجة لتعريف نوع فرعى وليكن .D فإنه ينبغى البدء بنفس هيكل الشىء للنوع الفرعى D كما هو للنوع الأصلى C وبنفس الطرق methods للنوع الفرعى D كما هو للنوع الأصلى C. ومن ثم يمكن تعديل النوع الفرعى D كالآتى:

١- لو أن هيكل النوع الأصلى C هو نوع سجل، أى يأخذ الشكل التالى:

RECORD OF (T1, ... Tk)

فإنه يمكن إضافة محتويات إضافية لهيكل السجل.

٢- قد تنشأ طرق جديدة تطبق فقط على النوع الفرعى D.

٣- قد يعاد تعريف البرامج الخاصة بالنوع الأصلى C لكى تملك معنى فى النوع الفرعى D.

ويلاحظ أن الأشياء التى هى أعضاء النوع الأصلى C هى وقائع له، كل منها له هيكل بيانات لذلك النوع ، وبرامج للنوع C يمكن أن تطبق لأى واقعة للنوع الأصلى C. وتكون الأنواع مرتبة هرميًا. أما فى حالة ما إذا كان النوع C نازلاً من النوع D عندئذ يمكن (عادة) تطبيق الطرق الخاصة بالنوع D على وقائع النوع C ولكن ليس العكس.

الطرق (البرامج) Methods :

يعرف كل برنامج لنوع Class خاص ويطبق لوقائع ذلك النوع ، ووقائع أنواعه الفرعية (لو وجدت). ويكون تعريف شكل البرنامج Method كالآتى:

```
method <Class name >
  < message format >
  < body of method >
%
```

اسم النوع < Class name > يمثل النوع المخصص له البرنامج method. وتعتبر < message format > اسمًا للبرنامج. واسم البرنامج وأسماء معاملاته Parameters (إن وجدت) وتمثل كتلة البرنامج < body of method > التشفير Code الذى يتم تنفيذه عندما يتم استدعاء البرنامج.

إنشاء أنواع السجلات Creating Record Types :

تسمح لغة الأوبال OPAL بتعريف أنواع Classes للأشياء Objects فى مصنفات مختلفة:

الحقائب bags (Multisets)، المنظومات arrays ، السلاسل الحرفية Strings وغيرها. وسوف يتم التركيز على الأنواع الآتية:

١- "الأشياء" المستعملة كأنواع سجلات.

٢- "الفئات" التى يمثل أعضاؤها أشياء لنوع ثابت.

وهذه الأنواع تناظر المنشآت Constructors التى تسمى "سجل من" RECORD OF و "فئة من" SET OF. ولإنشاء نوع لعلاقة Relation معينة مثلًا ، يعرف النوع C1 من النوع رقم (١) الذى تكون وقائعه هى مجموعات القيم المرتبة Tuples وإنشاء النوع C2 من النوع رقم (٢) لكى يكون النوع للعلاقة نفسها ، أى أنها فئة الأشياء للنوع C1. وتلك العلاقة قد تكون فقط واقعة النوع C2.

يوجد برنامج method مثبت بعدد من المعاملات ذات دالة Function تستخدم لتعريف نوع جديد new class. والمعلمات الثلاثة المهمة لبرنامج إنشاء النوع Class-Creation هى:

١- النوع الفرعى Subclass: اسم النوع < Class name > وقيمة هذه المعلمة هى اسم النوع الجديد.

٢- أسماء متغيرات الوقائع InstVarNames: اسم الحقل < field names > ، وهى تمثل أسماء متغيرات للأشياء التى تنتمى إلى النوع وقيمة هذه المعلمة هى أسماء الحقول. وتسمى متغيرات الوقائع لأنها تحدث لكل واقعة فى النوع الواحد.

٣- القيود Constraints : أنواع البيانات الأساسية < data types > ويوضع هذا القيد على النوع الذى له قيمة أو أكثر تنتمى إلى متغيرات وقائعه.

ويتم توصيف متغيرات الوقائع باستخدام منظومة ثوابت السلاسل الحرفية ، ومثل هذه المنظومة يتم فصلها بواسطة علامة الشباك "#" وكذلك القيود أيضًا يشار إليها كمنظومة ذات عناصر مزدوجة تحتوى على رمز الشباك "#" والذى اسم متغير الواقعة مسبوق بالعلامة "#" واسم النوع مثل الأنواع المثبتة كالسلاسل الحرفية String والأرقام الصحيحة integer.

إنشاء أنواع الفئات Creating Set Types :

لتعريف نوع Class سيسلك كفئة Set ، ترسل رسالة النوع الفرعى Subclass إلى فئة النوع المثبتة built-in. وهذه الأنواع الفرعية للفئة سوف لاتملك متغيرات وقائع ولكن تحتوى على أشياء Objects لنوع class واحد فقط.

وتسمح لغة الأوبال OPAL للفئات بأن تملك أشياء من الأنواع الأساسية Types المختلطة. وطبقًا لتعريف الفئة فإنه يمكن الحصول على الشكل التالى :

Set

Subclass : < Set name >

Constraints : < element Class >

معالجة البيانات Data Manipulation :

توفر قواعد البيانات الخاصة بمجموعة مهام قواعد البيانات DBTG ونظم إدارة المعلومات IMS أوامر "ابحث عن" FIND و "احصل على" GET على التوالى لعمل البحث والاسترجاع من قاعدة البيانات بالإضافة إلى توفير عمليات الإضافة والحذف والتعديل. وفى لغة الأوبال OPAL معظم العمليات الأمرية يجب أن توصف لكل نوع Class يتم تعريفه.

الإضافة Insertion :

بفرض أن S هى النوع Class المعرف لكى يكون فئة من T التى تمثل مجموعة القيم المرتبة Tuples. ومن ثم لو أردنا إنشاء واقعة للنوع S (ولتكن r مثلاً) لتخدم كعلاقة relation من النوع T ؛ لذا يتم إنشاء r بإرسال الرسالة "new" للنوع S كالآتى :

r := S new

ويمكن تعريف البرنامج method الخاص بالإضافة insert كما فى الشكل رقم (٧-١٥) وفيه يلاحظ ان "Insert" لا تظهر كاسم للبرنامج؛ ولذا يتم استعمالها فقط كاسم شكلى.

شكل رقم (٨ – ١٥) يوضح برنامج إضافة "Insert" للأصناف "Item"

```
1- method : Item Set
2- Insert Name : na
3- Insert Number : num
4- New Item
5- New Item :- Item Type new.
6- New Item Store Name : na ;
   Store Number : num.
7- Self add : New Item
```

ويستخدم البرنامج method معلمتين Two Parameters هما:

Insert Number , Insert Name.

ويلاحظ فى السطر رقم (٤) عبارة NewItem وهو أسلوب يتبع لتعريف المتغير المحلى للبرنامج وهذا المتغير فى هذه الحالة هو NewItem .

أما السطر (٥) فيتم جعل المتغير NewItem كشىء للنوع ItemType.

والسطر (٦) يضع متغيرات الوقائع للقيم المرغوبة na , num التى تمثل قيمًا للمعلمات السابقة. ويلاحظ وجود برنامجين مختلفين هما StoreName , StoreNumber يتم تطبيقهما للشىء NewItem.

أما السطر الأخير (٧) فيضيف قيمة مرتبة Tuple جديدة للفئة باستخدام المستقبل Self للبرنامج Add.

الاسترجاع Retrieval :

تداول قاعدة البيانات يتم الحصول عليها باتباع المسارات المضمنة فى هيكل مختلف الأنواع. على سبيل المثال : أحد متغيرات الوقائع لكل عميل (شىء) هى الطلبات Orders التى تقيد (أى يوضع عليها قيد) لكى تكون من نوع OrderSet ، أى فئة الطلبات Orders. بإرسال رسالة GetOrders للشىء من نوع CustType يمكن

الرجوع return بهذه الفئة من الطلبات ، وفعلياً يتم الحصول على مؤشر Pointer لتمثيل هذه الفئة وذلك يؤدى إلى تنفيذ عملية الاسترجاع بشكل أرخص بدون الإجبار على عمل نسخ لهذه الفئات التى قد تكون ذات بيانات كثيرة. وتستخدم لغة الأوبال OPAL هذا المؤشر لرؤية الشىء (يتم الفئة). وفى قاعدة بيانات الأوبال OPAL فإن المؤشرات للطلبات Orders تظهر فى الشىء (أى فئة الطلبات)، وإن فئة الطلبات لا تظهر مادياً فى سجل العميل Customer بل فقط مؤشراً للشىء (قيم الفئة).

برنامج الاختيار Select :

توفر لغة الأوبال OPAL طرقًا عديدة لاكتشاف كل أعضاء الشىء الممثل لفئة ما. إحدى هذه الطرق يتم باستخدام البرنامج method ذى المعلمة المسماة "اختر" Select. ومعلمة «اختر» هى كتلة للتشفير مع متغير محلى فردى، وشكل هذه الرسالة هو:

```
Select : [ :X  |< block of code > ]
```

حيث إن X تمثل المتغير المحلى وتستخدم كتلة التشفير black of Code كمعلمة لبرنامج اختر Select. على سبيل المثال: إذا رغبنا فى الحصول على فئة العملاء ذات الرصيد balance الأقل من صفر، فإن الفئة الجزئية للعملات ذات الأرصدة السالبة يمكن أن تخصص للمتغير الجديد D كالآتى:

```
D := Customers Selsct : [:C: | C Getbalance) < O  ]
```

برنامج الكشف Detect :

ويمكن إرسال فئة S إلى الرسالة "اكتشف" Detect المتبوعة بكتلة ذات المعلمة الواحدة. فعلى سبيل المثال: إذا رغبنا فى الحصول على الشىء للعميل Akram Salah فيمكن أن نرسل فئة العملاء للمتغير AS كالآتى:

```
AS : = Customers Detect :

[ : C | (C Get Name) = "Akram Salah"]
```

الحذف Deletion :

تحتاج لغة الأوبال OPAL بوصفها شيئًا موجهًا إلى أن يتم البحث عن الأشياء المراد حذفها قبل حذفها؛ لذا فمن الأفضل أن يتم تخزين هذه الأشياء فى متغيرات

تستعمل لهذا الغرض؛ وذلك لأن لغة الأوبال OPAL ليس لديها مؤشرات منتشرة تشير إلى هذه الأشياء تلقائيًا. لو كانت قيمة المتغير O هى شىء فى الفئة S عندئذ ترسل إلى الفئة S الرسالة التالية:

S remove : O

عندئذ سيتم حذف ذلك الشىء من الفئة S.

برنامج التنفيذ DO :

برنامج التنفيذ DO يطبق كتلة التشفير لكل عنصر للفئة ، وهى كتلة ذات المعلمة الواحدة. ويأخذ برنامج التنفيذ DO الشكل التالى:

DO : [: X | < Code involving X >]

كشافات الهوية Identity Indices :

يمكن إنشاء كشافات على أجزاء فرعية لعناصر الفئة حتى لو لم تكن هذه العناصر أنواعًا مستقلة بذاتها elementary. وتبنى هذه الكشافات على هوية الشىء للأشياء الموجودة فى هذه الحقول. والمسارات التى تشير إلى مثل هذه الحقول هى لها نفس الشكل $I_1, I_2, ..., I_n$. على سبيل المثال: لو فرض أن الفئة S هى فئة ذات العناصر للنوع C. وبفرض أن الأشياء للنوع C لديها متغير I ذات قيم ، عليها قيود لكى تكون من نوع ذى قيم قابلة للمقارنة مثل الأرقام وسلاسل الحروف (أنواع مستقلة بذاتها). عندئذ يمكن إنشاء كشاف على S بإرسال رسالة له ، هى:

S CreateEqualityIndex On : 'I'

ولو كانت هذه العناصر غير مستقلة بذاتها فإنه يمكن إنشاء كشاف على S بأرسال الرسالة التالية:

S CreateEqualityIndex On : I_1 . I_2 . ... I_n

مثال (٧-٣) :

بفرض أن لدينا قاعدة بيانات علاقية تحتوى على المخططات العلاقية relations التالية:

```
EMPLOYEE (EMP # , ENAME , JOB)
      PRIMARY  KEY (EMP #)

COURSE (COURSE # , TITLE)
      PRIMARY  KEY (COURSE #)

OFFERING (COURSE # , OFF # , ODATE , LOCATION)
    PRIMARY  KEY (COURSE # , OFF # )
    FOREIGN  KEY (COURSE #) References Course
ENROLLMENT (COURSE #, OFF # , EMP # , GRADE)
  PRIMARY  KEY (COURSE # , OFF # , EMP # )
  FOREIGN  KEY ( COURSE # , OFF # , EMP # )
  FOREIGN  KEY ( EMP #) References EMPLOYEE

TEACHER (COURSE # , OFF # , EMP #)
  PRIMARY  KEY (COURSE # , OFF # , EMP #)
  FOREIGN  KEY ( COURSE # , OFF # , References OFFERING
  FOREIGN  KEY ( EMP #) References EMPLOYEE
```

أ- تعريف البيانات فى لغة الأوبال OPAL :

١- تعريف النوع "الموظف" EMPLOYE المقابل للمخطط العلاقى EMPLOYEE فى السطور التالية :

```
1- OBJECT  SUBCLASS :  "EMPLOYEE"

2- InstVarNames          : #  [EMP," ENAME "," JOB"]

3- Constraints           : #  [#emp# , String],
                         [#ENAME, String],
                         [#JOB, String].
```

السطر رقم (١) يعرف نوع الشيء "الموظف" EMPLOYEE والنوع الفرعي sub-class للنوع المثبت المسمى OBJECT ، ويرسل هذا السطر رسالة Message إلى الشيء المسمى OBJECT الذى يسأله لكى ينفذ البرنامج المسمى Subclass وتصف عبارتا Constraints , InstVarNames المعلمات لذلك البرنامج. وتعرفان نوع جديداً new class يشابهما فى كل شيء يتم بإرسال رسالة إلى الشيء Object.

السطر رقم (٢) يبين أشياء النوع "الموظف" EMPLOYEE الذى لدية ثلاثة متغيرات وقائع عامة هى JOB , ENAME , EMP# .

السطر رقم (٣) يضع قيد لمتغير الواقعة #EMP لكى يحتوى على أشياء من نوع السلاسل الحرفية String وكذلك السطران الباقيان رقم (٤) ، (٥).

(١) تعريف النوع "المنهج" COURSE المقابل للمخطط العلاقى COURSE كالاتى:

```
1- OBJECT SUBCLASS  :  "COURSE"
2- InstVarNames    : # ["COURSE", "TITLE"," OFFERING"]
3- Constraints                    :  #  [#COURSE #, String],
                                        [#TITLE , String],
                                        [# OFFERING, OSET]].
```

السطر رقم (٥) يصف متغير الواقعة "فرص الدراسة" Offerings الذى سيحتوى على شيء من نوع OSET. وبشكل أكثر دقة يصف هوية الشيء (Object IDentity) OID للنوع OSET. وأن فرص الدراسة Offerings سوف تشير إلى فئة لكل الفرص الدراسية Offerings للمنهج Course فى سؤال. بمعنى آخر، لنمذجة علاقة الربط "منهج - فرص الدراسة بواسطة هرمية منع الانتشار Containment hierarchy تجعل الفرص الدراسية تحتوى بداخلها على المنهج المناظر.

(٢) تعريف النوع "فرص الدراسة" OFFERING المقابل للمخطط العلاقى OF-FERING كالآتى:

1- OBJECT SUBCLASS : "OFFERING"

2- InstVarNames : #["OFF # "," Odate "," Location",

3- " "Enrollments "," Teachers]"

4- Constraints : #[#[# Off # , String],

5- [# ODate , DateTime],

1- –[# Location, String],

2- –[# Enrollments , NSET] .

3- –[# TEACHER , TSET]].

ويلاحظ أنه ليس هناك اضطرار لاحتواء المفتاح الخارجى على متغير واقعة داخل الأشيـاء لنوع "فرص الدراسة" OFFERING للإشـارة إلى أشيـاء النوع "المنهج" COURSE .

(٣) تعريف النوع "التسجيلات" ENROLLMENT المقابل للمخطط العلاقى EN-ROLLMENT كالآتى:

1- Object Subclass : "ENROLLMENT"

2- InstVarNomes : # ["Emp "," Grade"]

3- Constraints : # [# Emp , EMPLOYEE].

[# geade , string]] .

وأيضًا الأشياء الخاصة بالنوع "التسجيلات" ENROLLMENT لا تحتوى على متغير واقعة للمفتاح الخارجى للإشارة إلى الأشياء الخاصة بالنوع "فرص الدراسة" OFFERING. متغير الواقعة EMP فى السطر رقم (٣) يحتوى على هوية الشىء OID للشىء من النوع "الموظف" EMPLOYEE الذى يمثل موظفًا employee فرديًا يختص

بتسجيل نفسه enrollment. ويلاحظ أن هرمية منع الانتشار تفسر ما يتعلق بكون الشىء «الموظف» موجوداً فعليًا داخل الشىء "يسجل نفسه" بالطبع. ولكن يلاحظ عدم التماثلية حيث إن "التسجيلات" Enrollments هى علاقة ربط متعدد – لمتعدد ، ولكن مساهمتين فى علاقتى الربط "يتيح فرص الدراسة Offerings و"يوظف" Employees ويتم معالجتها بشكل مختلف تماماً.

(٥) تعريف النوع "المدرس" TEACHER المقابل للمخطط العلاقى TEACHER ومعالجته كنوع فرعى لنوع "الموظف" EMPLOYEE يتم كالآتى:

```
1- EMPLOYEE SUBCLASS   : "TEACHER"
2- InstVarNames        : # [ " COURSES"]
3- Constraints         : # [#[# COURSE . CSET].
```

يعرف السطر الأول النوع "المدرس" TEACHER بأنه نوع فرعى لنوع "الموظف" EMPLOYEE المعرف من قبل. بمعنى أن "المدرس" TEACHER "يكون" ISA "موظف" EMPLOYEE وهكذا كل شىء "مدرس" Teacher له متغيرات وقائع هى JOB , ENAME # EMP , كل المورثة من "الموظف" Employee بالإضافة إلى النوع "المنهج" COURSE الذى سيحتوى على هوية الشىء OID للشىء من النوع CSET. وسوف يتم تعريف تجميعات "Collection" نوع الشىء للخمس الأنواع المعرفة أعلاه وهى:

ESET , CSET , OSET , NSET , TSET

* على سبيل المثال فإن الشىء للنوع ESET سوف يتكون من فئة من OID للأشياء الفردية لنوع "الموظف" EMPLOYEE ويتم تعريفها كالآتى:

```
1- SET Subclass : "ESET"
2- Constraints   :   EMPLOYEE.
```

السطر رقم (١) يعرف نوع الشىء ESET كنوع فرعى للنوع المثبت SET.

السطر رقم (٢) يضع قيد على الأشياء من النوع ESET لكى تكون فئات لهوية الأشياء من نوع "الموظف" EMPLOYEE. ويلاحظ أن أشياء النوع ESET ليس لديها متغيرات وقائع عامة ، وكذلك بقية الفئات.

SET Subclass : "CSET"

Constraints : COURSE

Set Subclass : "OSET"

Constraints : "OFFERING:"

SET Subclass : "NSET"

Constraints : "ENROLLMENT

SET Subclass : "TSET"

Constraints : TEACHER

(ب) معالجة بيانات فى لغة الأوبال OPAL:

*** إنشاء فئة خالية لهوية الأشياء:**

فى حالة تجميع هوية أشياء OIDs كل الموظفون فى الشىء ESET. ينبغى أولاً إنشاء الشىء ESET كالآتى:

EM-OID : = ESET NEW

والتعبير فى الجانب الأيمن لهذا التخصيص يعود بهوية شىء OID جديدة ، أى بفئة فارغة لهوية أشياء "الموظف" Employee أى أن النوع ESET خالٍ من الوقائع. وبصفة عامة فإن المتغير EM سوف يستعمل لتعريف الفئة لكل الموظفين employees.

*** إضافة هوية الشىء بالفئة:**

عند إنشاء شىء جديد "للموظف" فإن هوية ذلك الشيئ تضاف فى الشىء ESET بواسطة هوية الشىء OID المخزنة فى المتغير EM-OID.

أ- لذا يتم تعريف برنامج Method لإنشاء مثل هذا الشىء الخاص بالموظف Em-ployee وإضافة هويته فى الشىء ESET :

```
1- METHOD      : ESET
2- ADD-EMP #     : EMP#-PARM                /* المعاملات */
3- ADD-NAME        : ENAME-PARM
4- ADD-JOB    : JOB-PARM
5- |EMP-OID|                               / * متغير محلى */
6- EMP-OID    : EMPLOYEE NEW         / * * موظف مبدئى * /
7- EMP-OID SET-EMP #    EMP #-PARM  / * تخصيص مبدئى * /
8-
9- SET-JO              : JOB-PARM
10- SELF ADD          : EMP-OID.            /* إضافة */
```

(ب) وبهذا يكون لدينا برنامج لإضافة موظف جديد لقاعدة البيانات كالآتى:

```
EM-OID    ADD-EMP#           : E003
          ADD-ENAME    " :JAMAL"
          ADD-JOB       " :PROGRAMMER"
```

وهو ما يناظر فى لغة الاستعلام البنائية SQL لقواعد البيانات العلاقية الشكل الآتى:

```
INSERT INTO EMPLOYEE (EMP # , ENAME , JOB)
: (PROGRAMMER"" ,          VALUES ("E003" , JAMAL"
```

عمليات الاسترجاع :

فى حالة البحث عن كل فرص الدراسة Offerings بمدينة الرياض Riyadh للمنهج C100. ويقصد من المثال وجود متغير XX ذى قيمة هى هوية الشيء OID لفئة كل المناهج؛ ومن ثم يمكن وضع التشفير كالآتى:

```
1- | COURSE-C 100 , C 100-OFFS , C100-NY-OFFS |
2- COURSE-C 100
3- : = XX DETECT : [ : CX |
                              " C100 = "CX GET-COURS #| .
4- C 100-OFFS
5-                     : = COURSE-C100   GET-OFFERINGS.
6- C100-NY-OFFS
7-                     : = C100-OFFS SELECT : [ :0X |
                              "RIYADH =  "OX GET-LOCATION ]
8- –^ C100-NY-OFFS.
```

١- السطر رقم (١) يصف ثلاثة متغيرات محلية هي:

* COURSE-C100 الذى سيستخدم لتخزين الهوية الشيء OID للمنهج C100.

* C100-OFFS الذى سيستخدم لتخزين الهوية الشيء OID لفئة كل الفرص الدراسية OFFERINGS.

* C100-NY-OFFS الذى سيستخدم لتخزين الهوية الشيء OID لفئة هوية الاشياء للفرص الدراسية المطلوبة.

٢- السطر (٢) ، (٣) يرسل رسالة message إلى المشار إليه بواسطة المتغير .XX وتستدعى الرسالة البرنامج المثبت DETECT لكى يتم تطبيقة على ذلك الشيء.

٣- السطر (٤) ، (٥) يخصص لهوية الشيء "لفئة كل فرص الدراسة " للمنهج C100 للمتغير C100-OFFIS.

٤- السطر (٧) ، (٧) مثل السطر (٢) ، (٣) للبرنامج المثبت SELECT الشبيه بالبرنامج المثبت DETECT عدا عودته بهوية الشيء OID لفئة هوية الأشياء OIDs الخاصة بكل الأشياء. بذلك التأثير تخصص هوية الشيء لفئة هوية الأشياء لفرص الدراسة بمدينة الرياض للمنهج C100-NY-OFFS.

٥- السطر (٨) يعود بهوية الشيء OID والرمز ^ يستخدم للعرض Display أو الناتج العائد Return Result.

*** عمليات التحديث:**

١- التعديل:

وتتم بنفس طريقة الاسترجاع عدا أن برامج SET-V يستعمل بدلاً من برامج GET-V.

٢- الحذف:

يستخدم معه البرنامج المثبت REMOVE لحذف الشيئ على سبيل المثال:

```
EM-OID REMOVE : DETECT : [: EX|
                "E200" = EX GET-EMP#] .
```

وبهذا يتم حذف الموظف E200 من فئة كل الموظفين EMPLOYEES.

الهوامش :

1- [Hallock, 1998], Patrick Hallock, 'Composite Objects in Relational and Object Relational Constructors Using InfoModelle ', **Journal of Conceptual Modeling**, April, 1998.

2- [Stonebraker, 1998], Michael Stonebraker, Paul Brown and Dorothy Moore, **Object-Relational DBMSs**, Second Edition, The Morgan Kaufmann Publishers, Sept. 1998.

3- [Kroenke, 1999], David M Kroenke, 'Database Processing Fundamental, Design, and Impementaion, Seventh Edition, Prentice Hall,1999.

4- [Barry, 1998], Douglas Barry, **Standard Overview ODMG 2.0**, The Morgan Kaufmann Publishers, April 1998.

5- [Cattell, 2000], R.G.G. Cattell and Douglas K Barry, **The Object Data Standard ODMG 3.0**, The Morgan Kaufmann Publishers, 2000.

6- [GRHAM, 1991], Ian Grham, **BIS Applied Systems: Object-Oriented Models**, Addison-Wesley Publishing Company. Inc., 1991.

7- [RIAD, 1994], Mokhtar B. Riad and Saber Abd Allah, 'Object-Oriented Databases: Features, Capabilities, Products and Development Trends '. The 19th International Conference for Statistics, Computer Science, Scientific and Social Applications, Cairo, 9-14 April, 1994.

8- [DATE,1995], C.J. Date, An **Introduction to Database Systems**, Volume I, Sixth Edition, Addison-Wesley Publishing Company Inc.,1995.

9- [ULLMAN,1988], Ullman J.D., **Principles of Database and Knowledge-base System**', Vol. 1, Computer Science Press, 1988.

الفصل التاسع

التحويل بين نماذج قواعد البيانات التقليدية

مقدمة :

على الرغم من تزايد شعبية قواعد البيانات العلاقية إلا أن قواعد البيانات التبحرية (الهرمية والشبكية) تظل فعالة فى معالجة برامج المعاملات القوية والبنوك والمكتبات وشركات التأمين وخطوط الطيران وغيرها من مستخدمى الأحجام الضخمة للبيانات لذلك لا يزالون يعتمدون على استخدام نظم قواعد البيانات الهرمية مثل نظم إدارة المعلومات IMS أو نظم قواعد البيانات الشبكية مثل النموذج التشاورى للغة نظام البيانات CODASYL. وإنه من الأسهل أن تبرمج التطبيقات فى بيئة علاقية ولكن العديد من المؤسسات لا ترغب فى إنفاق ملايين الدولارات للتحويل من نظم قواعد البيانات التبحرية إلى نظم قواعد البيانات العلاقية. ومع ذلك تغير الحاسبات أو نظم إدارة قواعد البيانات قد يستلزم مثل هذا التحويل وهو ما يحدث الآن من تحويل فى بيئات الحاسبات الضخمة إلى بيئات شبكات الحاسبات الشخصية . وكذلك ما بدا واضحاً فى تعديل بعض التطبيقات التى تعمل فى طى نظم قواعد البيانات التبحرية لمشكلة عام ٢٠٠٠ وما تطلبه من إنفاق ملايين الدولارات . ولازالت قواعد البيانات العلاقية معدلها أقل فى الأداء مقارنة مع نظم قواعد البيانات التبحرية . ويمثل تحويل قواعد البيانات مجالاً مهماً فى البحث العلمى . وسوف نتناول فى هذا الفصل العمل فى هذا المجال حيث يمكن وضع الأسس العلمية للتحويل بين نموذج كينونة علاقة المطورEER والنماذج التقليدية و كذلك بين نماذج قواعد البيانات الهرمية، والشبكية ، والعلاقية فيما بينهما[١]. ويقدم هذا الفصل الموضوعات التالية:

التحويل المنطقى لنموذج كينونة – علاقة المطور EER إلى النماذج التقليدية:

يشمل هذا التحويل النماذج التقليدية الثلاثة (العلاقية – الشبكية – الهرمية) كل بشكل منفرد. ويتم تقسيم هذا التحويل فى كل نموذج إلى مرحلتين. أولاهما: تشمل قواعد التحويل من نموذج كينونة – علاقة ER إلى نموذج قاعدة البيانات التقليدى المراد التحويل إليه. ثانيتهما: تشمل قواعد التحويل المنطقى من نموذج كينونة – علاقة المطور EER إلى النموذج التقليدى المحدد المراد التحويل إليه. حيث يتضمن نموذج كينونة – علاقة المطور EER العديد من المفاهيم الإضافية الخاصة ، والتى سبق التطرق

إليها فى الفصل السادس مثل: التعميم والتخصيص، النوع الفرعى المشارك، والمصنفات.

التحويل بين نماذج قواعد البيانات التقليدية:

يتضمن هذا التحويل مرحلتين. تشمل المرحلة الأولى على التحويل المنطقى بين هياكل نماذج البيانات ، سواء من الهياكل التبحرية (الهرمية - الشبكية) إلى العلاقية أو بالعكس. فى حين تشتمل المرحلة الثانية على التحويل المادى لقاعدة البيانات والتى تأخذ بعين الاعتبار تنظيمات الملفات وقرارات طرف التداول.

أمثلة على التحويل بين نماذج قواعد البيانات التقليدية:

سوف يتم تقديم العديد من الأمثلة على التحويل المنطقى بين نماذج البيانات التقليدية المختلفة إلى جانب تقديم التحويل المادى لقاعدة البيانات الناتج عن عملية تعديل أو تطوير نتائج تصميم قاعدة البيانات المادى لأغراض الأداء.

علاقات الربط متعدد - لمتعدد للنماذج التقليدية:

يتم استعراض مثال توضيحى لعلاقات الربط متعدد - لمتعدد عن التحويل المادى لقاعدة البيانات ، حيث تكون هذه العلاقات معقدة إلى حد ما فى نظام إدارة المعلومات IMS لقواعد البيانات الهرمية، وكذلك فى النظام التشاورى للغة نظام البيانات CODASYL لقواعد البيانات الشبكية. فى حين تتجه إلى سهولة التصميم فى قواعد البيانات العلاقية.

التحويل المنطقى لنموذج كينونة - علاقة المطور EER إلى النماذج التقليدية:

أولاً : التحويل المنطقى لنموذج كينونة - علاقة المطور EER إلى النموذج العلاقى:

(أ) قواعد التحويل من نموذج كينونة - علاقة ER إلى النموذج العلاقى

١- بفرض أن نوع الكينونة L ذات الخصائص A (L) حيث إن

$A(L) = \{ k_1, L_1 \ldots\ldots L_n \}$

وأن الخاصية K_1 هى المفتاح الأساسى لنوع الكينونة L وإن نوع الكينونة L ينبغى يتم تحويلها إلى جدول علاقى يسمى L له نفس خاصية المفتاح الأساسى K1.

٢- وأن نوع الكينونة R ذات الخصائص A (R) حيث إن:

$A(R) = \{ k_2, r_1, \ldots r_m \}$

يحول نوع الكينونة R إلى الجدول العلاقى R له نفس خاصية المفتاح الأساسى K2.

٣- فى حالة علاقة الربط واحد لواحد:

بفرض أن الجدول العلاقى L كان لابد من تقسيمه إلى جدولين علاقين هما L_1, L_2 طبقًا لتبعيات الوظيفية، حيث إن الجدول العلاقى L1 ذى الخصائص A (L1) كالآتى :

$A(L1) = A(L_1, L_2, \ldots L_{i-1})$

وإن الجدول العلاقى L2 ذى الخصائص A(L2) كالآتى:

$A(L2) = \{Li, Li+1, \ldots Ln\}$

فإن كلا الجدولين العلاقيين يصبحان كالآتى:

$L_1 = A(L_1) \cup K_1$

$L_2 = A(L_2) \cup K_1$

أى لا بد من إلحاق المفتاح الأساسى K_1 إلى كلا الجدولين العلاقيين.

٤- في حالة علاقة لربط واحد - لمتعدد:

بفرض وجود علاقة ربط واحد - لمتعدد بين نوع الكينونة L ونوع الكينونة R. ففى تلك الحالة يمكن وضع المفتاح الأساسى لنوع الكينونة L فى نوع الكينونة R وبذلك يصبح الجدول العلاقى R كالآتى:

$R= \{ k_2 , r_1 ,\ldots r_m \} \cup \{k1\}$

حيث يشكل k1 مفتاحًا خارجيًا للجدول العلاقى R ويصبح المفتاح الأساسى الجديد هو المفتاح المكون من الخاصيتين k2 ، k1.

٥- فى حالة علاقة الربط متعدد - لمتعدد:

لو فرض أن علاقة الربط بين نوعى الكينونة R ، L فى الخطوة الأولى والثانية كانت متعدد - لمتعدد ، فلا بد من إنشاء جدول علاقى جديد وليكن X ذا الخصائص A(X) ، حيث إن هذه الخصائص تمثل تقاطع البيانات بين كلا الجدولين العلاقين؛ ومن ثم فإن :

$A(X) = A(L) \cap A(R)$

ويصبح الجدول العلاقى الجديد X كالآتى:

$X= A(X) \cup \{K1\} \cup \{K2\}$

حيث إن كلاً من مفتاح K1 ، K2 مفتاح خارجى يشكلان معًا المفتاح الأساسى للجدول العلاقى الجديد الناشئ.

ب- قواعد التحويل المنطقى من نموذج كينونة علاقة المطور EER إلى النموذج العلاقى:

يتم تحويل المفاهيم الأخرى الإضافية الخاصة بعملية تطور Enhancement وتحويل نموذج كينونة - علاقة ER إلى نموذج كينونة - علاقة المطور EER إضافة إلى القواعد المتبعة فى تحويل نموذج كينونة - علاقة ER إلى النموذج العلاقة الموضحة فى الخطوة (أ). وفيما يلى قواعد تحويل المفاهيم الإضافية الخاصة بنموذج كينونة - علاقة المطور EER كالآتى:

(١) التخصص/التعميم:

بفرض أن تخصيصًا له عدد m نوع فرعي {S1 , S2 ، Sm} ويعمم بالنوع الأصلى C حيث إن خصائص النوع C هى {K , a1 a2 , ... aj} وتمثل الخاصية K المفتاح الأساسى N مخطط العلاقى باستعمال الاختيارات التالية:

(أ) إنشاء جدول علاقى P للنوع الأصلى C بالخصائص A (P) حيث إن

$A (P) = \{K , a_1 , a_2 ,... a_j\}$

حيث تمثل الخاصية K المفتاح الرئيسى للجدول العلاقى P . وأيضًا يجب إنشاء جدول علاقى Pi لكل نوع فرعى Si (حيث إن 1? i m) وتشمل الخصائص التالية:

A (Pi) = {K} U | Si خصائص}

وهذا يعنى إضافة المفتاح الأساسى K الخاص بالجدول العلاقى P إلى كل جدول علاقى فرعى Pi .

(ب) إنشاء جدول علاقى Pi لكل نوع فرعى Si (حيث إن 1 ? I ? m) مع الخصائص. ويصبح الجدول العلاقى بالخصائص A (Pi) كالآتى:

A (Pi) = { Si خصائص} U { K , $a_1, a_2, ... a_j$}

وتمثيل الخاصية K المفتاح الأساسى للجدول العلاقى Pi ويلاحظ من مخطط الجدول العلاقى التكرار الزائد عن الحد الذى سوف يطرأ على تخزين البيانات.

(جـ) إنشاء الجدول العلاقى P بكل خصائص النوع الأصلى والأنواع الفرعية المرتبطة به كالآتى :

A(P)= {K.a1,a2,... aj} U {S1 خصائص } U ... U {Sm خصائص} U{t}

ويستخدم هذا الاختيار للأنواع الفرعية المنفصلة disjoint. وتمثل t الخاصية التى تشير إلى النوع الفرعى الذى له قيم مرتبة tuple تنتمى له.

(د) إنشاء جدول علاقى واحد P بالخصائص التالية :-

A (P) = {K.a1,....aj} U { S1 خصائص U ... U { Smخصائص} U {t1,t2,...tm}

وهذا الاختيار للتخصيص ذات الأنواع الفرعية المتداخلة ومنفصلة وكل ti هى خاصية منطقية تشير إلى ما إذا كان الصف ينتمى إلى النوع الفرعى Si أم لا ، حيث إن m ?i ?1. ويحتمل أن ينشئ هذا الاختيار عدداً من القيم الخالية Null.

مثال ذلك تحويل عدد من الأنواع الفرعية التى تشكل معًا تخصيصًا معينًا أو التى تعمم النوع الأصلى مثل الأنواع الفرعية "للموظف" EMPLOYEE التى سبق ذكرها مثل "السكرتارية" SECRETARY و "الفنيين" TECHICIAN و"المهندسين" ENGINEER، وفيما يلى الاختيارات الشائعة لتمثيل التخصيص[٢]:

- إنشاء المخطط العلاقى للنوع الأصلى Superclass (فى التعميم) بكل خصائصه بما يتضمنه من مفتاح أساسى. وأيضًا إنشاء المخططات العلاقية لكل نوع فرعى Subclass (فى التخصيص) والتى يتضمنها مع إضافة المفتاح الأساسى الذى يتضمنه النوع الأصلى إلى كل نوع فرعى.

- كما هو موضح بالشكل رقم (٩-١) المناظر لمخطط نموذج كينونة - علاقة المطور EER فى الشكل رقم (٥-٢٣).

- وتتم عملية الربط بناءً على المفتاح الأساسى بين أى مخطط علاقى لأى نوع فرعى مع المخطط العلاقى الخاصة بالنوع الأصلى للحصول على كل الكينونات المحددة والخصائص المورثة لهذه الكينونات فى أى نوع فرعى.

شكل رقم (٩-١) يوضح كيفية تحويل التخصيص (أوالتعميم) فى نموذج كينونة - علاقة المطور لمخططات علاقية

EMPLOYEE

SSN	Fname	Mname	Lname	Bdate	Addr	Job Type

SECRETARY

SSN	TSpeed

TECHNICIAN

SSN	TGrade

ENGINEER

SSN	Eng Type

(٢) النوع الفرعى المشارك:

* النوع الفرعى المشارك مثل "مدير قسم الهندسة" Engineering-Manager فى الشكل رقم (٥-٢٥) هو نوع فرعى لعديد من الأنواع الأصلية. وهذه الأنواع يجب أن يكون لها جميعها نفس المفتاح الأساسى. بخلاف ذلك فإن النوع الفرعى المشارك ينبغى أن ينمذج كمصنف Category. يتم إنشاء مخطط علاقى لكل نوع أصلى بكل خصائصه بما يتضمنه من مفتاح أساسى. ويتم كذلك إنشاء مخطط علاقى للنوع المشارك.

* بالخصائص التى يتضمنها مع إضافة المفتاح الأساسى الذى يتضمنه كل نوع أصلى إلى النوع المشارك. كما هو موضح بالشكل رقم (٩-٢) المناظر لمخطط نموذج كينونة - علاقة المطور EER فى الشكل رقم (٥-٢٥). ويمكن تطبيق أى من الاختيارات السابقة، وعادة يستعمل الاختيار (أ).

شكل رقم (٩-٢) تحويل النوع المشارك "مدير قسم الهندسة" كنوع فرعى لعديد من الأنواع الأصلية كما فى شكل (٥-٢٥)

MANAGER

SSN	NAME	ADDR

ENGINEER

SSN	NAME	ADDR	Eng-Type

Salared-Employee

SSN	Hours	Rates	Salary

Engineering -Manager

SSN	Project	Locaton

المصنفات Categories :

المصنف هو نوع فرعى لاتحاد اثنين أو أكثر من الأنواع الأصلية وهذه الأنواع ذات مفاتيح مختلفة بسبب اختلاف أنواع الكينونات . مثال ذلك : المصنف "صاحب" OWNER فى الشكل رقم (٥-٢٦) الذى يمثل فئه جزئية لاتحاد ثلاثة أنواع كينونات هى "الشخص" PERSON و"البنك" BANK و"الشركة"COMPANY التى لها مفاتيح مختلفة. وقد تكون الأنواع الأصلية للمصنف لها نفس المفتاح ، مثال ذلك: المصنف "مسجل المركبة" المركبة" Register-Vehicle فى الشكل رقم (٥-٢٦).

* بخصوص المصنف الذى يتم تعريفه باستخدام أنواع أصلية مختلفة المفاتيح ، ينبغى عند إنشاء المخطط العلاقى الذى سيتم استخدامه لمناظرة المصنف ،أن يخصص له مفتاح جديد يسمى "المفتاح النائب" Surrogate key. مثال ذلك: إنشاء المخطط العلاقى "صاحب" OWNER لمناظرة المصنف "صاحب" OWNER كما هو موضح بالشكل رقم (٩-٣)، واشتماله على خصائص للمصنف فى هذا المخطط. المفتاح الأساسى ("المعرف" OwnerID) للمخطط العلاقى "صاحب" هو المفتاح النائب. ويضاف هذا المفتاح إلى كل مخطط علاقى مناظر للنوع الأصلى للمصنف.

* أما بخصوص المصنف الذى يعرف باستخدام أنواع أصلية لها نفس المفتاح فلا حاجة لإنشاء المفتاح النائب بل يستخدم نفس المفتاح الشائع فى الأنواع الأصلية كما هو مبين فى الشكل رقم (٩-٣).

الشكل رقم (٩-٣) تحويل المصنفات فى شكل رقم (٥ - ٢٦) إلى مخططات علاقية

PERSON

SSN	Driver License No	Name	Addr	Owner ID

BANK

Bname	Baddr	Owner ID

COMPANY

CName	Caddr	Owner ID

OWNER

Owner ID

Registered-Vehicle

VehicledId	License Plate No

CAR

Vehicled ID	CStyle	Cmake	CModel	CYear

TRUCK

Vehicled ID	Tmake	Tmodel	Tonnage	TYear

OWNS

OwnerID	VehicledID	Purchase Date	Lien OrRegular

ثانيًا : التحويل المنطقى من نموذج كينونة علاقة المطور EER إلى النموذج الشبكى :

(أ) قواعد التحويل من نموذج كينونة - علاقة ER إلى النموذج الشبكى:

(١) يتم تحويل كل نوع كينونة إلى نوع سجل.

(٢) يتم تحويل علاقة الربط التى تمثل واحد - لمتعدد إلى نوع فئة بين نوع سجل الأب ونوع أعضاء السجل.

(٣) يتم تحول علاقة الربط التى تمثل متعدد - لمتعدد إلى علاقات ربط واحد - لمتعدد باستخدام نوع سجل الوصل Juncture record type.

(٤) الرسم التخطيطى للمخطط الشبكى، ونوع الفئة يتم تمثيلها بسهم يشير من نوع سجل الأب إلى نوع أعضاء السجل.

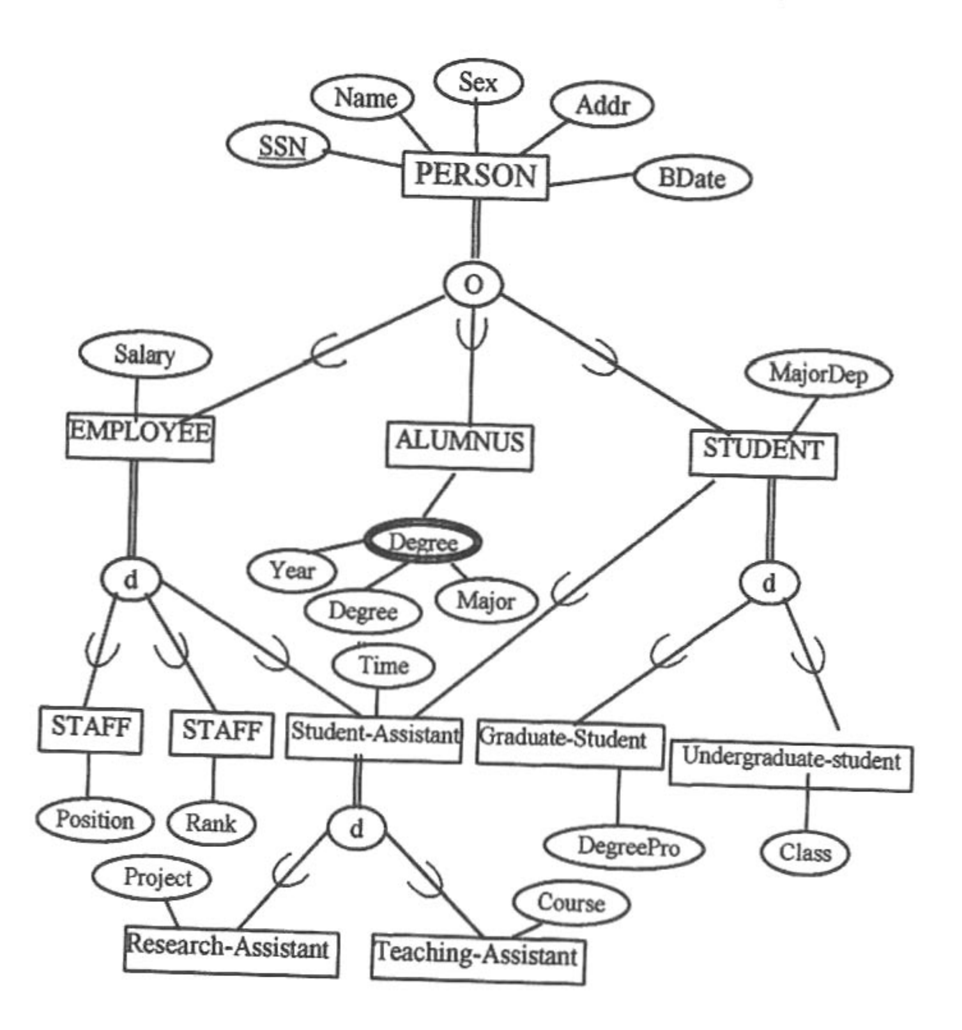
(ب) قواعد التحويل المنطقى من نموذج كينونة - علاقة المطور EER إلى النموذج الشبكى :

(ا) التخصيص و التعميم :

يمكن استخدم الاختيارات (أ)، (جـ)، (د) المناظرة للحالات العلاقية والفارق الرئيسى هو لو أن علاقة الربط واحد - لواحد ، فإن أنوا ع الفئات Set types تستعمل بدلاً من تكرار خاصية المفتاح فى الاختيار (أ). وعلى الرغم من أن الاختيار (ب) لم ينشئ نوع السجل للنوع الأصلى على قدر الإمكان، إلا أنه نادراً ما يستعمل؛ لأن علاقة الربط واحد - لواحد يمكن نمذجتها طبيعياً بنوع الفئة. وعلاقة الربط نوع الأصلى/فرعى تذكر بأن نوع الفئة واحد - لواحد فيها كل قيمة الفئة التى لها على الأكثر سجل واحد عضواً. ونظراً لأن هذا القيد ليس جزء من النموذج الشبكى ، فإن برامج التطبيق يجب أن تلتزم به عند استعمال نظم إدارة قواعد البيانات الشبكية. وسوف يستعمل لنوع الفئة واحد - لواحد علاقة الربط "يكون" IS-A: لأن سجل كل عضو يمثل نفس الكينونة مثل نوع كينونة الأب لكن فى وظيفة النوع الفرعى. كل أنوا ع الفئات هذه ينبغى أن توضح إجبارياً لتجبر علاقة الربط "يكون" IS-A.

* يستخدم الشكل رقم (٩-٥) تشابكية التخصيص لنوع شخص Person / { موظف Employee ، طالب student { الموضحة فى الشكل رقم (٩-٤)، ويلاحظ اختيارياً أنه يمكن تكرار خانة (حقل) بيانات المفتاح من نوع السجل الأب إلى نوع السجل العضو ويتم توصيف القيد الهيكلى على نوع الفئة. وفى هذا المثال يمكن تكرار خانة بيانات المفتاح "رقم التأمين الاجتماعى SSN من نوع سجل الشخص PERSON فى أنوا ع السجلات الأعضاء "الموظف EMPLOYEE و"الطالب" STUDENT [٢].

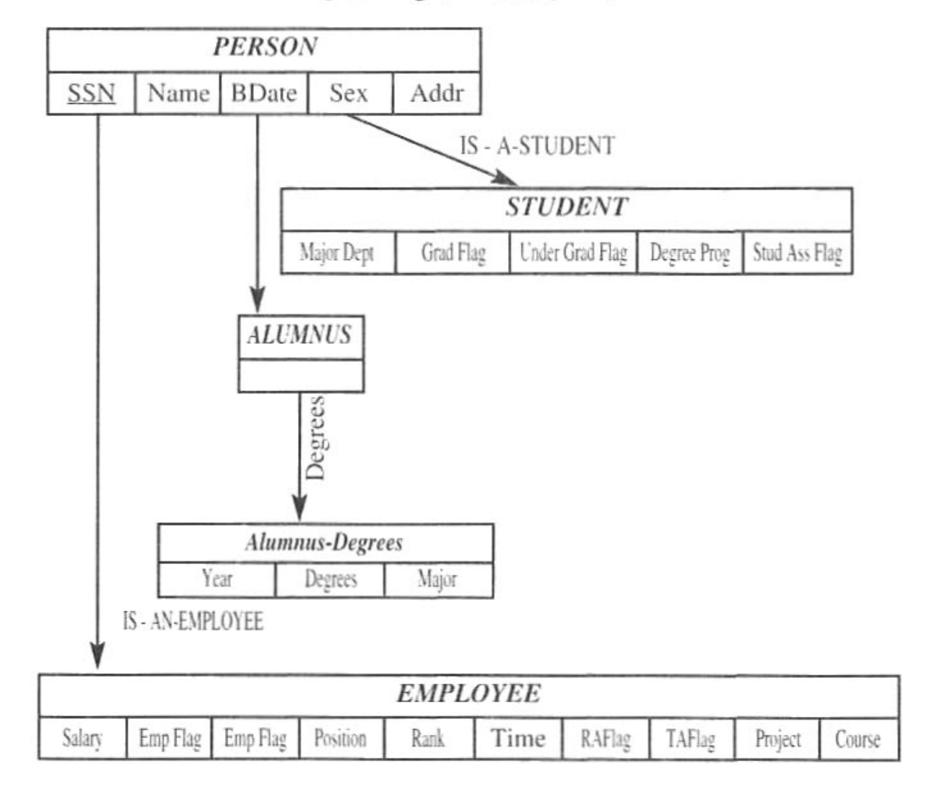
الشكل رقم (٩-٤) تخصيص هرمى/تشابكى لقاعدة بيانات الجامعة الافتراضية



(٢) النوع الفرعى المشارك :

يستعمل الاختيار (أ) لتحويل النوع الفرعى المشارك مثل "شخص" Person نوع السجل الذى يمثل النوعى الفرعى المشارك ينبغى ان يكون عضو فى العديد من علاقات الربط واحد - لواحد (١ : ١) التى تمثل أنواع الفئة الإجبارية "يكون" IS-A [٢] كما هى موضحة بالشكل (٩- ٥).

شكل رقم (٩-٥) تحويل مفاهيم نموذج كينونة علاقة المطور EER فى الشكل رقم (٧-٤) إلى النموذج الشبكى



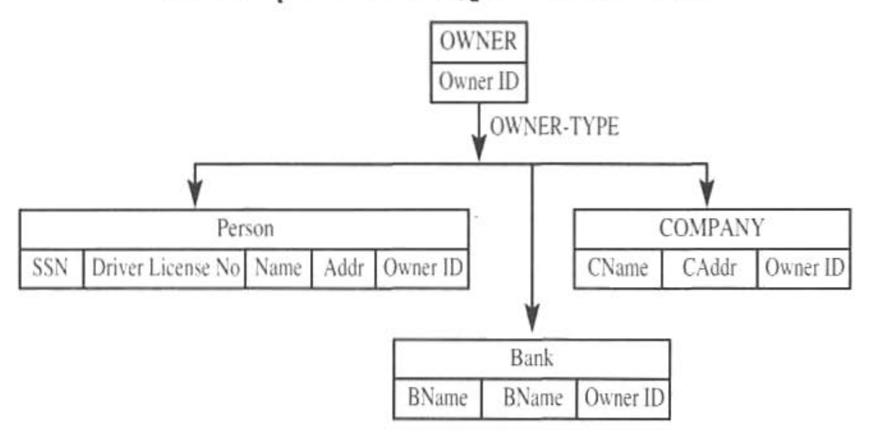
(٣) المصنفات :

هناك العديد من الاختيارات لتحويل المصنف إلى النموذج الشبكى. لو كانت أنواع الكينونات منفصلة disjoint (صفة معظم المصنفات) قد يتم استعمال نوع كينونة أب واحد لنوع فئة عضو متعدد ، حيث إن نوع سجل الأب يمثل المصنف. يجب إجبار القيد الإضافى على أن كل سجل فى نوع سجل الأب يكون منفصلاً كسجل عضو واحد فى أنواع سجلات الأب. وهناك اختياران لتحويل المصنف من نموذج كينونة - علاقة المطور EER إلى النموذج الشبكى[٢]:

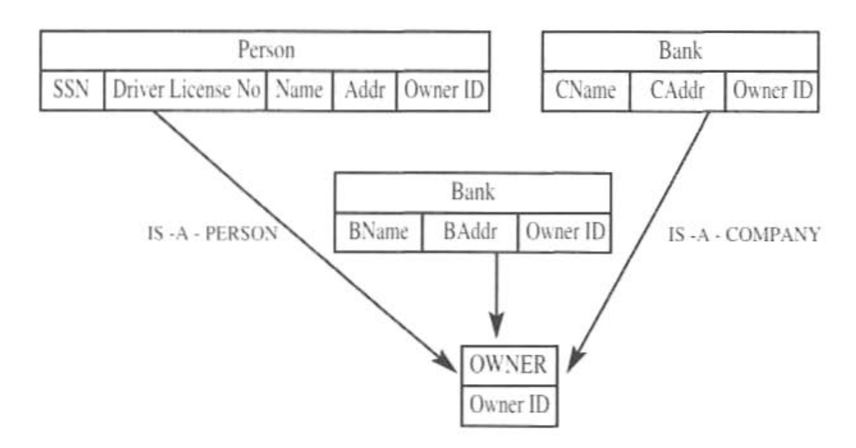
١- لو أن تعريف أنواع الكينونات كان منفصلاً فإنه يمكن استعمال نوع الفئة المتعددة الأعضاء لأب واحد ، حيث إن نوع سجل الأب يمثل المصنف ويجب أن يتم إجباره على إضافة القيد الذى يبين أن كل سجل فى نوع سجل الأب متصل بالضبط بسجل عضو من أنواع السجلات الأعضاء. على سبيل المثال المصنف "صاحب" Owner فى الشكل رقم (٥-٢٦) هو فئة جزئية لاتحاد ثلاثة أنواع كينونات منفصلة هى"الشخص" PERSON و "الشركة" COMPANY و"البنك" BANK ، وتم تحويلها إلى فئة متعددة الأعضاء كما هو مبين بالشكل رقم (٩-٦أ).

٢- يمكن جعل المصنف عضواً فى العديد لأنواع الفئة الاختيارية واحد - لواحد (١:١)، وذلك عندما يتم تعريف نوع أصلى يكون أباً لواحد من هذه الأنواع. ويوضح الشكل رقم (٩-٦ب) تمثيل المصنف "صاحب" OWNER بالإضافة إلى قيد أن كل سجل فى المصنف "صاحب" يجب أن يكون عضواً فى واحد من أنواع الفئة التى يجب أن يتم الإلتزام بها. ويعتبر حقل "المعرف" OwnerID اختيارياً ويمكن أن يلتزم بالقيود الهيكلية.

شكل رقم (٩-٦أ) تحويل المصنف إلى نموذج شبكى باستعمال نوع فئة العضو المتعدد OWNER-TYPE لتمثيل المصنف "صاحب" OWNER



شكل رقم (٩-٦ب) تحويل المصنف إلى نموذج شبكى باستعمال ثلاثة أنوا ع فئات لتحويل المصنف "صاحب" OWNER



ثالثاً : التحويل المنطقى من نموذج كينونة علاقة - المطور EER إلى النموذج الهرمى:

(١) نوع أصلى/ فرعى :

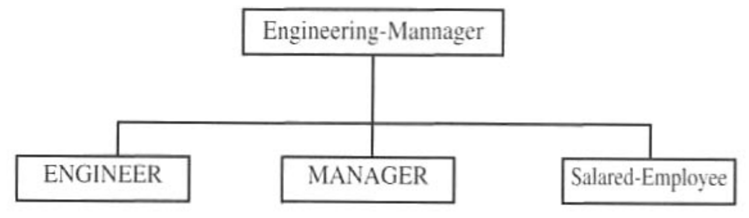
علاقة الربط نوع أصلى/فرعى يمكن أن تنمذج كعلاقة ربط أب - ابن (واحد - لواحد)، حيث إن كل نوع سجل أب له على الأكثر نوع سجل ابن واحد وبرامج التطبيق يجب أن تجبر هذا القيد عند تطبيق النظام الهرمى.

(٢) النوع الفرعى المشارك :

* يمكن أن يحول النوع الفرعى المشارك كعلاقة ربط أب - أبن واحد - لواحد (١:١) والعديد من علاقات الربط أب - ابن واحد - لواحد (١: ١) الافتراضية مع مشاركة النوع الفرعى كأب أو كابن افتراضى فى كل من هذه العلاقات.

* ونظراً لمحدودية فرضية أن السجل يمكن أن يكون له على الأكثر أب حقيقى واحد وأب افتراضى واحد ، فإن هذا يتم عمله فقط لنوع فرعى مشارك مع اثنين من الأنواع الأصلية كما هو مبين فى الشكل رقم (٩-٧) للنوع الفرعى المشارك "مدير قسم الهندسة" Engineering-Manager لمخطط نموذج كينونة - علاقة المطور EER[٢].

شكل رقم (٩-٧) يوضح تحويل الأنواع الفرعية المشاركة إلى النموذج الهرمى. يحول النوع الفرعى المشارك كأب للأنواع الأصلية.



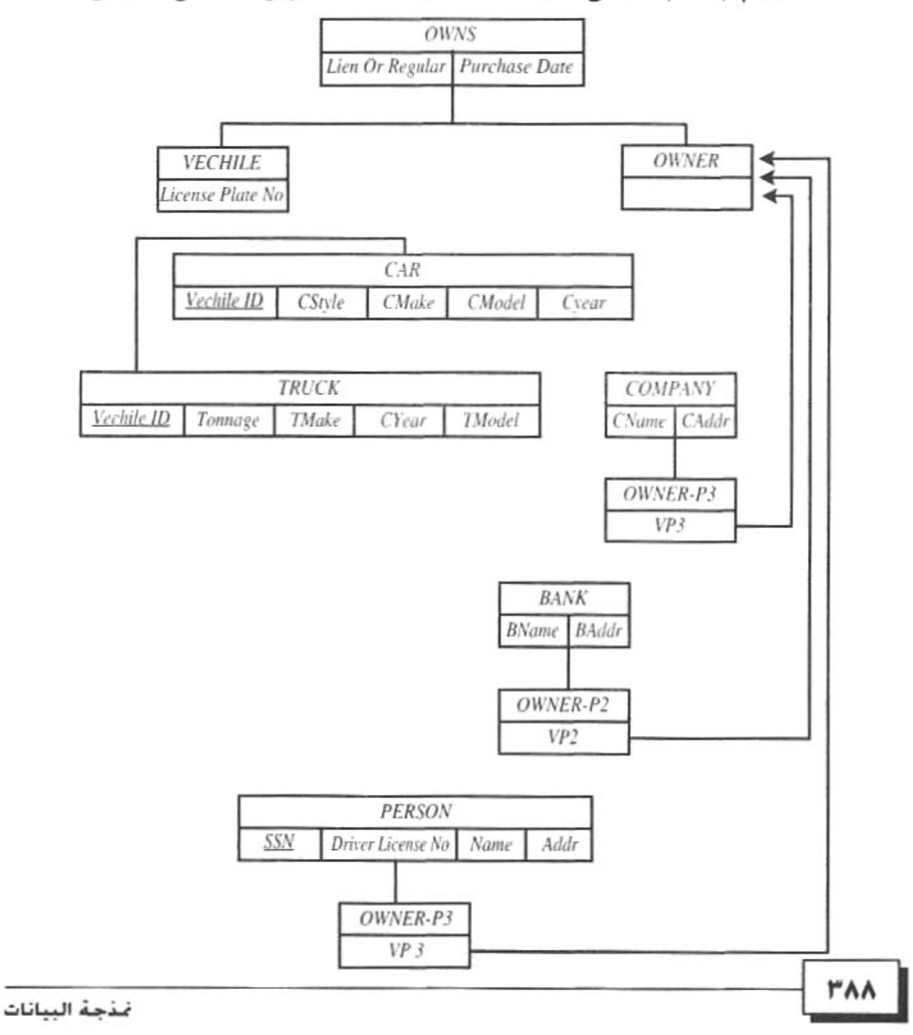
(٣) المصنفات :

يمكن توصيف اختيارين لتحويل المصنف. ولتوضيح هذين الاختيارين يمكن استخدام الشكل رقم (٥-٢٦) لتحويله إلى المخطط الهرمى ، حيث يمكن تمثيل نوع علاقة الربط "يملك" Owns كنوع سجل الأصل root وهذان الاختياران هما[٢]:

١- يكون المصنف هو نوع سجل الأب، وعلاقة الربط أب - ابن واحد - لواحد (١:١) يتم تعريفها لكل نوع سجل يمثل تعريفًا لنوع المصنف . كما هو مبين فى الشكل رقم (٩-٨) للمصنف "مركبة" .VECHLE ويلاحظ أن هذا الاختيار يعمل فقط مع المصنف التام.

٢- يمكن إنشاء نوع سجل مؤشر تحت كل نوع سجل يناظر تعريف النوع الأصلى للمصنف.

شكل رقم (٩-٨) يوضح طريقتين لتحويل المصنفات إلى النموذج الهرمى.



التحويل بين نماذج قواعد البيانات التقليدية :

١- التحويل المنطقى بين هياكل نماذج البيانات :

تصميم قواعد البيانات المنطقى هو عملية تجميع حقول بناء على الرباطات بينهم ، وتصمم قاعدة البيانات لهذه الحقول التى تتطابق مع القواعد الهيكلية لنظام إدارة قواعد البيانات الخاصة. يوجد العديد من طرق تصميم قواعد البيانات المنطقية جيدة التعريف ، المبنية على تقنية تطبيع البيانات، طريقة كينونة – علاقة ومنهجية ديت ولسون Date-wilson تقنية تطبيع البيانات ترتبط بشكل طبيعى مع قواعد البيانات العلاقية. وهو إجراء ذو تعريف جيد لترتيب الحقول فى مجموعات أو جداول أو علاقات لها صفات معينة. عندما يتم معالجة هذه المجموعات كملخصات هيكلية لملفات خطية وتعبأ هذه الملفات ببيانات فعلية ، والتكرار فى هذه البيانات ليس له تأثير يذكر، حيث إن أهمية علاقات الربط فى البيانات تكون محمية. فى أوائل الثمانينيات طور ديت ولسون تقنية بديلة لتصميم قواعد البيانات العلاقية. هذه التقنية افترضت أن الكينونات وتعريفها لحقولها التى لا تتكرر هى سهلة التعريف فى بيئة التطبيق حيث إنها تنتج قواعد بيانات صحيحة.

التحويل المنطقى من الهياكل التبحرية (الهرمية والشبكية) إلى العلاقية

: Logical Conversion from Novigational (hierarchic and network) to Relational structures

التحويل من قواعد البيانات التبحرية إلى قواعد البيانات العلاقية يتم مباشرة بالاعتماد على القواعد الآتية [١]،[٣]:

١- كل جزئية (نوع سجل) لنظام إدارة المعلومات (IMS) الهرمية أو كل نوع سجل للنموذج التشاورى للغة نظام البيانات CODASYL الشبكى الذى يمثل كينونة العالم الحقيقى real world ينبغى تحويله إلى جدول علاقى فى قاعدة البيانات العلاقية مع الخصائص المناظرة لحقول الجزئية أو نوع السجل Segment / record.

٢- كل الجداول العلاقية التى تم إنشاؤها فى الخطوة السابقة ينبغى أن تحتوى على حقول مضافة تمثل المفاتيح الخارجية لإنشاء علاقات الربط واحد – لمتعدد التى يتم تمثيلها كفروع فى الهياكل التبحرية حسب الاحتياج لاكتمال معرفاتهم identifiers

الوحيدة (التى لا تتكرر). والمفاتيح الخارجية هى حقول للجزيئات Segment الأعلى أو لأنواع السجلات الأعلى فى النماذج الهرمية أو الشبكية على التوالى.

٣- كل جزئية ابن مزدوج منطقى لنظام إدارة المعلومات IMS logical child pair (الملف الافتراضى) أو نوع سجل الوصل juncture record فى النموذج التشاورى للغة نظام البيانات CODASYL الذى ينشئ علاقة الربط متعدد - لمتعدد ينبغى أن يحول إلى جدول علاقى فى قاعدة البيانات العلاقية.

٤- كل الجداول التى يتم إنشاؤها فى الخطوة السابقة ينبغى أن تتضمن حقولاً مضافة تمثل المفاتيح الخارجية لكى لا يتكررلنوعى كينونة علاقة الربط متعدد لمتعدد. وهذه الحقول التى يشار اليها كمفاتيح خارجية ، يجب تكون أيضًا حقول مفتاح الجزئيات أو أنواع السجلات فى فروع الهياكل التبحرية.

التحويل المنطقى من الهياكل العلاقية إلى الهياكل التبحرية

: Logical Conversion from Relational to novigational Structures

التحويل من قاعدة البيانات العلاقية إلى قاعدة البيانات التبحرية يحتاج إلى معرفة تامة لعلاقات الربط بين الجداول (العلاقية) . والقواعد الأساسية التى تحكم عملية التحويل تتبع كالآتى:

١- كل جدول علاقى فى قاعدة البيانات العلاقية يحتوى على مفتاح أساسى ينبغى أن يتم تحويله إلى جزئية أعلى Superior فى نظام إدارة المعلومات IMS أو إلى نوع سجل أعلى فى النظام التشاورى للغة نظام البيانات CODASYL.

٢- كل جدول علاقى يتضمن مفتاحًا خارجيًا ، يجب أن تفحص الخصائص التى تتضمن هذا المفتاح وتناظر المفاتيح الاساسية للجداول العلاقية الأخرى؛ لأنه ينبغى أن يتم تحويلها إلى جزئية تابعة Subordinate أو نوع سجل تابع فى النموذج الهرمى أو الشبكى على التوالى.

٣- فى حالة إسهام المفتاح الخارجى للجدول العلاقى فى الخطوة السابقة فى أكثر من جدول علاقى ، عندئذ ينبغى أن تكون علاقة الربط متعدد - لمتعدد. لذلك فإن هذا الجدول سوف يحول إلى ابن مزدوج منطقى لنظام إدارة المعلومات IMS (الملف

الافتراضى) أو نوع سجل وصل للنظام التشاورى للغة نظام البيانات CODASYL فى قواعد البيانات الهرمية أو الشبكية على التوالى.

٤- بخلاف ذلك فإن علاقات الربط سوف تكون واحد - لواحد أو واحد - لمتعدد. وهذا الجدول ينبغى أن يحول إلى جزئية تابعة فى نظام إدارة المعلومات IMS أو نوع سجل تابع فى النظام التشاورى للغة نظام البيانات CODASYL للجزئية العليا فى نظام إدارة المعلومات IMS أو لنوع سجل الأب فى النظام التشاورى للغة نظام البيانات CODASYL الذى تم الإشارة إليه فى الخطوة الأولى.

٢- التحويل المادى قاعدة بيانات :

■ فى التصميم المبدئى لقاعدة بيانات باستخدام نظام إدارة المعلومات IMS يجب أن يكون مأخوذاً بعين الاعتبار تنظيمان للملف الرئيسى وقرارات طرق التداول:

١- ما نوع التداول الموجود لجزئية الأصل Root Segment خلال مفتاحه الوحيد ؟

٢- كيف تخزن قيم الجزئية Segment occurrences التابعة بالنسبة لجزئية الأصل ؟

طرق التداول فى نظام إدارة المعلومات IMS : خيارات تداول جزئية الأصل تكون تسلسلية ، Sequantial ، مفهرسة Indexed ومفرومة hashed بينما خيارات تداول الجزئية التابعة تكون تسلسلية وذات مؤشر Pointer-bases.

(ب) قاعدة البيانات الشبكية فى النظام التشاورى للغة نظام البيانات تكون CO-DASYL، وأنوا ع السجلات الأعلى ترتيباً والتابعة (المعرفة كفئة عند التجميع) مباشرة متصلة كل بالأخرى بسلاسل دائرية Circular Chains للمؤشرات التالية. وقيم أى سجل خاص يمكن أن يتم استرجاعها خلال قيم السجلات الأعلى ترتيباً عن طريق المؤشرات التالية أو حيث تم تخزين المؤشرات ويتم استرجاعها خلال التفريم hashing.

(ج) الجداول العلاقية فى قواعد البيانات العلاقية ليس لها مؤشرات لتصل بمجموعات القيم المرتبة ولكن من خلال خصائص (حقول) المفاتيح.

بالإضافة إلى بعض النظم العلاقية مثل إنجرس INGRES تسمح بالتخزين والاسترجا ع للقيم المرتبة tuples من الجداول العلاقية بواسطة التفريم hashing.

أمثلة للتحويل بين نماذج قواعد البيانات التقليدية

: Examples of conversion between traditional Data base Models

التحويل من النماذج الشبكية إلى كل من النماذج الهرمية والعلاقية

: Conversion from Network to both Hierarchic and Relational Models

بالنسبة للنموذج الشبكى سوف نوضح كيفية التحويل للحالة المعقدة Complex Case لعلاقة الربط متعدد - متعدد . ويبين الشكل رقم (٩-٩) علاقة الربط متعدد – لمتعدد بين نوع سجل المدرس Teacher والطالب Student.

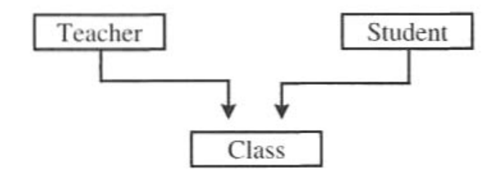
قاعدة البيانات الشبكية فى النظام التشاورى للغة نظام البيانات CODASYL، علاقة الربط متعدد لمتعدد بين المدرس Teacher والطالب Student يتم تنفيذها باستخدام نوع سجل الوصل juncture record (أو ما يعرف باسم ملف الوصل (Connection file. ونفرض فى هذه الحالة أن ملف الوصل هو ملف الفصل Class file. وفيه يتكون نوع السجل من الفئة المكونة له وسجل المدرس Teacher بالإضافة إلى الفئة المكونة له وسجل الطالب Student. وتمثل هذه الطريقة تفكيك النموذج الشبكى المعقد ذات علاقات الربط متعدد – لمتعدد إلى علاقات ربط واحد – لمتعدد. وعلاقات الربط واحد – لمتعدد يمكن تطبيقها باستخدام تقنيات تنظيم الملفات المعروفة مثل:

- الملفات متعددة الوصلات Multilink Files .
- أو الملفات المعكوسة Inverted Files .

والنموذج الشبكى البسيط هو مجموعة من علاقات الربط واحد – لمتعدد التى تنتج تمثيلًا هو الرسم الدائرى Cyclic graph كما فى الشكل رقم (٩-٩).

وفى الحقيقة فإن النموذج التشاورى للغة نظام البيانات System Language يسمح للمصمم أن يعرف أنواع السجلات المشتركة فى التقاطع intersection records (data) لكى تمثل كوصلة زائفة Pseudo link بين أنواع السجلات فى علاقة الربط متعدد – لمتعدد وفى نفس الوقت يتم تخفيض التكرار بين البيانات المترابطة.

شكل رقم (٩-٩) تمثيل علاقة الربط متعدد - لمتعدد فى النموذج الشبكى



التحويل من نموذج قاعدة البيانات الشبكى إلى نموذج قاعدة البيانات الهرمى

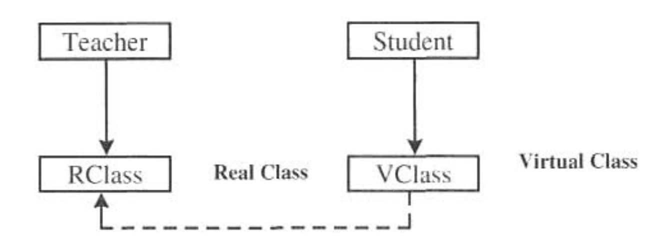
: Conversion From Network to Hierarchic Database Model

يتم تمثيل النموذج الهرمى وهيكل البيانات فى شكل شجرة. وإنه من الصعب جداً أن يلتقط الترتيب الهرمى عفوياً بأن يتم وضع نوع سجل المدرس Teacher كسجل أب ونوع سجل الطالب Student كسجل أبن. بهذا التمثيل البسيط يمكن البحث عن كل سجلات الطلبة لسجل مدرس معين. ولكن بسبب علاقة الربط مدرس - طالب متعدد - لمتعدد، فإن كل سجل طالب قد يتكرر لكل سجل مدرس حيث إن المدرس يدرس للطالب؛ لذا توجد مشكلتان بهذا التمثيل:

أولاً : ربما يوجد تكرار كثيراً للبيانات؛ مما يؤدى إلى تضاربها وعدم اتساقها.

ثانيًا: لايجاد كل سجلات المدرسين الذين يدرسون لطالب ما لابد من بحث كل قاعدة البيانات. شجرتان يمكن أن يستعملا فى إحداهما يكون نوع سجل المدرس Teacher أب Parent node ونوع سجل الطالب Student هو الابن Child node. بينما فى الشجرة الاخرى يكون نوع سجل الطالب هو الأب Parent node ونوع سجل المدرس هو الابن Child node. ولكن بهذا التمثيل تصبح مشكلة التكرار duplication أكثر سوء. ليس فقط تكرار سجل الطالب لكل مدرس حيث يقوم المدرس بتدريس الطالب ولكن سجل المدرس أيضًا يتكرر لكل سجل طالب حيث يدرس الطالب أيضًا لدى المدرس. وأحسن تمثيل يتم باستخدام الملف الافتراضى Virtual file، كما فى الشكل رقم (٩-١٠). حيث يحتوى الملف الافتراضى على سجلات افتراضية كمؤشرات Pointers للسجلات الفعلية . وبهذا يمكن حل مشكلة تكرار البيانات وتظل علاقة الربط الأصلية متماثلة.

شكل رقم (٩-١٠) الهيكل الهرمى لقاعدة البيانات مدرس - طالب باستعمال السجل الافتراضى



وإنه من المهم ملاحظة أن السجل الافتراضى يتطلب تداولاً أكثر بواسطة المؤشرات، وإنها فكرة جيدة أن يتم وضع السجلات الفعلية فى المكان الذى يسمح بتداولها بشكل أكثر تكراراً . ويمكن حساب عدد الأشجار المطلوبة لتمثيل ملفات قاعدة البيانات الهرمية بالمعادلة التالية:

$$K = ?\ (n_i - 1) + 1$$

حيث إن K تمثل عدد الأشجار الناتجة.

إن ni عدد الأباء لكل نوع كينونة لها أكثر من أب واحد.

إن n عدد الكينونات التى لها أكثر من أب.

فى الشكل رقم (٩-٩) ومثال على المعادلة السابقة حيث توجد نوع كينونة واحدة (سجل الفصل Class entity type) لها سجلان أبوان: سجل الأب الأول للمدرس والآخر للطالب؛ لذلك عدد الأشجار الذى يمثل هذه الملفات هو:

K = (2-1) +1 = 2 trees

ومن ثم لو تم تفكيك الشكل رقم (٩ - ٩) إلى الهيكل الهرمى ، فسوف يتم الحصول على شجرتين كما هو مبين بالشكل رقم (٩ - ١٠).

التحويل من نموذج قاعدة البيانات الشبكى إلى نموذج قاعدة البيانات العلاقية : Conversion from Network to Relational Database Model

كما هو معروف أن التمثيل العلاقى يسمح بنوع واحد فقط للأشياء هو الملفات التى يطلق عليها الجداول العلاقية relations. وهكذا لا توجد وصلات links صريحة بين الجداول العلاقية. ولكن كيف يمكن تمثيل علاقة الربط واحد – لمتعدد فى قاعدة البيانات العلاقية؟ الإجابة عن هذا، وقد سبق توضيحها فى الفصل الرابع، هى أن مثل العلاقات يتم تمثيلها ضمنيًا بواسطة خصائص الجداول العلاقية. وهذه الخصائص التى تربط بين أنواع السجلات الأعلى Superior وأنواع السجلات التابعة Subordinate هى حقول مفتاح لا تتكرر لأنواع السجلات العليا وهى تمثل المفاتيح الخارجية فى أنواع السجلات التابعة. كل سجل وصل Juncture record فى النظام التشاورى للغة نظام البيانات CODASYL الذى ينجز علاقات الربط متعدد – لمتعدد ينبغى أن يتحول إلى جدول علاقى فى قاعدة البيانات العلاقية. وكل من الجداول العلاقية التى سبق إنشاؤها ينبغى أن تتضمن حقولاً إضافية لكى تشير بشكل لا يتكرر إلى نوعى الكينونة فى علاقة الربط متعدد – لمتعدد. هذه الحقول التى هى أيضًا يشار إليها كمفتاح خارجى، ينبغى أن تكون حقولاً للسجلات العليا فى الهيكل الشبكى فى الخطوة الأولى ولتحويل علاقة الربط متعدد – لمتعد بين نوع سجل المدرس Teacher ونوع سجل الطالب Student فى الشكل رقم (٩–٩)، ولا بد من تحويل السجلات الاعلى لكل من المدرس والطالب إلى جداول علاقية بعد حذف الوصلات links الصريحة (المؤشرات المادية) كما فى شكل (٩–١١).

شكل رقم (٩–١١) المخططات العلاقية لقاعدة بيانات مدرس – طالب

TEACHER

TNo	*TName*	*Position*	*Dept*

STUDENT

SNo	*SName*	*Address*	*TelNo*	*Classification*

فى الخطوة الثانية، نوع سجل الفصل Class record الذى يمثل سجلات الوصل Juncture records ملف الوصل (Connection file) لعلاقة الربط متعدد – لمتعدد فى

قاعدة البيانات الشبكية ينبغى أن يتحول إلى جدول علاقى يحتوى على مفاتيح لا تتكرر من كلا الجدولين العلاقيين حيث تمثلهما خاصية TNo للمدرس وخاصية SNo للطالب. وهذه الخصائص ينبغى أن تضاف إلى الجدول العلاقى "فصل" لتمثل المفتاح الخارجي. ويمثل الجدول العلاقى "فصل" فى الشكل رقم (٩-١٥) علاقة الربط بين الجدولين العلاقيين المدرس والطالب[١].

شكل رقم (٩-١٢) علاقة الربط للمخططين العلاقيين مدرس و طالب

STUDENT

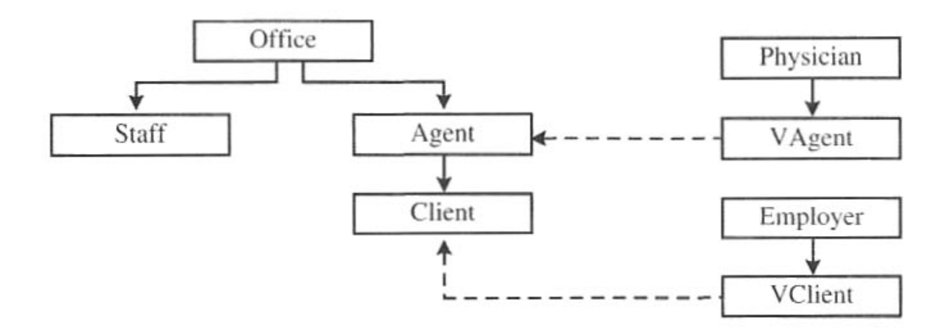
SNo	*SName*	*Address*	*TelNo*	*Classification*

٣- التحويل من النماذج الهرمية إلى كل من النماذج الشبكية والعلاقية

: Conversion From Hierarchic to both Network and Relational Models

يتركب مثال قاعدة البيانات الهرمية من ثلاثة أشجار كما فى الشكل رقم (٩-١٣)، وسوف يتم استخدام هذا المثال لتوضيح مفاهيم التحويل إلى كل من النماذج الشبكية والعلاقية. فى قاعدة البيانات الهرمية، علاقة الربط بين الجزئية الأعلى Superior Segment والجزئية التابعة Subordinate Segment هى واحد - لمتعدد كما هى مبينه فى الشكل رقم (٩-١٣) حيث تمثل علاقات الربط فى الشجرة الأولى جزئية "المكتب" Office ومجموعة العاملين Staff ، وبين جزئية المكتب Office والوكيل Agent وبين جزئية الوكيل Agent "والعملاء" Client. ولكن من المهم ملاحظة علاقة الربط بين جزئية الوكيل Vagent و"الطبيب" فى شجرة الطبيب - الوكيل Physicion-Vagent. والوكيل الافتراضى يمثل السجل الافتراضى الذى يحتوى على المؤشرات فقط التى تشيرإلى الجزئية الحقيقة (جزئية - الوكيل فى شجرة المكتب Office Tree) والذى يؤدى إلى تجنب تكرار نفس البيانات.

شكل رقم (٩-١٣) الهيكل الهرمى لشركة تأمين افتراضية ذات أشجار ثلاثة

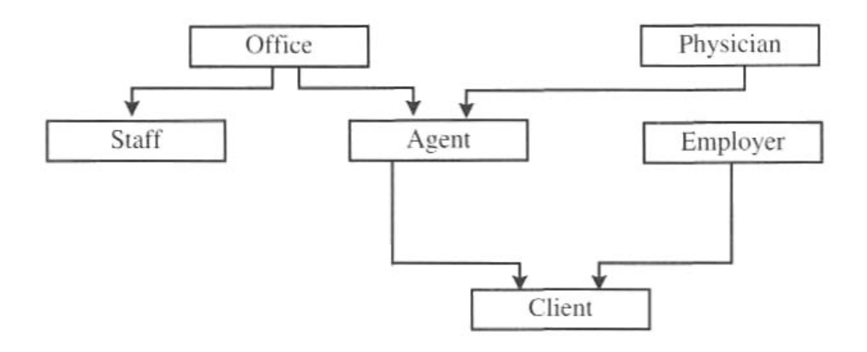


التحويل من نموذج قاعدة البيانات الهرمية إلى نموذج قاعدة البيانات الشبكية

: Conversion From Hierarchic to network Database Model

إنه من السهل التحويل بين نماذج قواعد البيانات التبحرية فيما بينها. وهذا ما اتضح من المثال السابق الخاص بالمدرس – الطالب Teacher - Student عن كيفية التحويل من رسم graph إلى شجرة tree فى علاقة الربط متعدد – لمتعدد حيث ملف الوصل Connection File له أكثر من سجل أب. فالملف الحقيقى Real File الذى تم اختياره أبًا هو الذى يكون تداوله بشكل أكثر تكرارًا فى حالة التحويل. نموذج قاعدة البيانات الهرمية ينبغى ألا يحتوى على بيانات زائدة عن الحد. أما عن الآباء الآخرين الذين يشاركون فى ملف الوصل الذين تم تمثيلهم فى أشجار أخرى ، يلحق بكل منها ملف افتراضى. وإنه من السهل اتباع نفس الخطوات السابقة فى اتجاه مغاير. وهذا يعنى أن أى ملف افتراضى مثل ملف Vagent سوف يربط مع ملفه الحقيقى فى شجرة أخرى وينشئ ملف الوصل؛ لكى يمثل علاقة الربط متعدد – لمتعدد كما فى الشكل رقم (٩-١٤). أما علاقات الربط الأخرى واحد – لواحد ، واحد – لمتعدد تظل كما هى دون تغيير.

شكل رقم (٩-١٤) نموذج قاعدة البيانات الشبكية لشركة التأمين الافتراضية



التحويل من نموذج قاعدة البيانات الهرمية إلى نموذج قاعدة البيانات العلاقية

: Conversion from Hierarchic to Relational Database Model

إنه من الأهمية بمكان معرفة علاقات الربط بين الملفات فى قاعدة البيانات الهرمية قبل التحويل إلى نموذج قاعدة البيانات العلاقية. فى حالة علاقة الربط البسيطة واحد - لمتعدد حيث تمثل الجزئية العليا بجدول علاقى يحتوى على كل الخصائص كما هى. ويجب أيضًا تمثيل الجزئيات التابعة حيث المفتاح الخارجى الذى تمت الإشارة إليه فى التحويل من نموذج قاعدة البيانات الشبكية إلى نموذج قاعدة البيانات العلاقية؛ لذلك فإن تمثيل الحالة المعقدة الذى تمثل علاقة الربط متعدد-لمتعدد تحتاج إلى خطوتين[1].

الخطوة الأولى: حيث علاقة الربط متعدد-لمتعدد لا توجد بشكل صريح فى قاعدة البيانات الهرمية لكنها تمثل ضمنيًا مع الاتجاهية المزدوجة بين الملفين الحقيقى والافتراضى.

الخطوة الثانية: يجب الجمع بين كل من الملفات الافتراضية والحقيقية فى جدول علاقى واحد مع المفتاح الخارجى؛ لأنه يتكون من المفاتيح الرئيسية لملفات الآباء. ونموذج قاعدة البيانات العلاقية الذى ينتج يمكن تلخيصه فى الشكل رقم (٩-١٥).

الشكل رقم (٩-١٥) نموذج قاعدة البيانات العلاقية لشركة التأمين الافتراضية

OFFICE

OAddr	OTel No	Manager

STAFF

OAddr	SName	SAddr	Position

PHYSICIA

PName	PAddr	PTel No

AGENT

OAddr	PName	AName	AAddr	Commission

EMPLOYER

Comp Name	Comp Addr

CLIENT

OAddr	AName	Comp Name	CName	CAddr	PL Type	PL Num

التحويل من نماذج علاقية إلى كل من النماذج الشبكية والهرمية

: Conversion From Relational to both Network and Hierarchic Models

لا توجد فى النموذج العلاقى مؤشرات صريحة بين الجداول العلاقية كما هو الحال فى النماذج التبحرية. يصف مخطط قاعدة البيانات العلاقية هيكل قاعدة البيانات، حيث يعطى اسمًا للجدول العلاقى وأسماء للخصائص. والعديد من الخصائص لكل جدول علاقى وكل خاصية لها نوع أساسى مرتبط بها يشير إلى نطاق القيم المسموح بها لتلك الخاصية. إن المرحلة الضرورية للتحويل من النماذج العلاقية إلى هياكل قاعدة البيانات التبحرية هو أن يدرس بعناية الربط بين الجداول العلاقية ، حيث يمكن استنتاج الرسم التخطيطى الذى يبين علاقات الربط (واحد - لواحد ، واحد - لمتعدد، متعدد - لمتعدد) بين الجدول العلاقية كما فى الشكل رقم (٩-١٦) الذى يعبر عن مثال توضيحى ويكشف عن مخطط قاعدة البيانات العلاقية لإدارة

التخزين Depertment Store. ومن مثال إدارة التخزين يمكن استنتاج الرسم التخطيطى فى الشكل رقم (٩-١٧).

شكل رقم (٩-١٦) قاعدة بيانات علاقية لإدارة التخزين

Department

D Name	Location	TelNo

Employee

DName	SSNo	Name	CName	Addr	Salary

Sales

DName	SSNo	AccNo	Receipt No	Date	Item name	Quantity	Amount

Customer

Acc No	Name

التحويل من نموذج قاعدة البيانات العلاقية إلى نموذج قاعدة البيانات الشبكية : Conversion From Relational to Network Database Model

يتضح من الشكل رقم (٩-١٦) أن علاقة الربط بين جدول الإدارة Depertment وجدول الموظفين Employee هى علاقة ربط واحد – لمتعدد فى حين أن جدول المبيعات Sale يمثل علاقة ربط متعدد – لمتعدد.

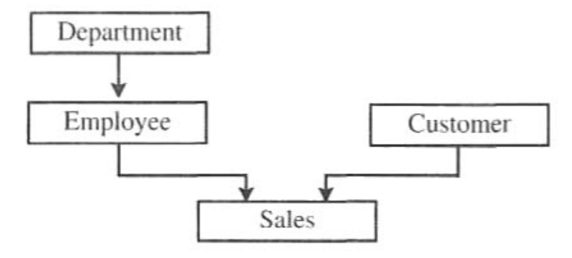
الخطوة الأولى:

فى علاقة الربط واحد – لمتعدد يتم حذف المفتاح الخارجى من جدول الموظفين Em-ployee ووضع المؤشر فى سجل الأب الخاص به لكى يشير إليه. حيث إن جدول الموظفين يمثل نوع السجل التابع وجدول الإدارة Department هو نوع السجل الأعلى. كما أنه فى قواعد البيانات الشبكية فى النظام التشاورى للغة نظام البيانات CO-DASYL تتصل أنوا ع السجلات العليا والتابعة بشكل مباشر لكل واحد مع السلاسل الدائرية للمؤشرات. يمكن استرجا ع القيم لأى نوع سجل خاص خلال قيم سجلاتهم العليا عبر المؤشرات اللاحقة.

الخطوة الثانية :

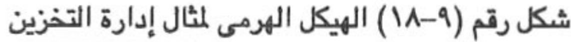
بعد فحص الرسم التخطيطي المستنتج يظهر جدول للمبيعات الذي يمثل ملف وصل ذا مؤشرين، كل منهما يشارك بواسطة مفتاحه الأساسي لتكوين المفتاح الخارجي. كما حدث في الخطوة الأولى بحذف المفتاح الخارجي ووضع المؤشرات بين أنوا ع السجلات العليا (العميل Customer والموظفين Employee) ونوع السجل التابع (المبيعات Sale) كذلك في الحالة المعقدة لتمثيل علاقة الربط متعدد - لمتعدد.

شكل رقم (٩-١٧) الهيكل الشبكي لمثال إدارة التخزين

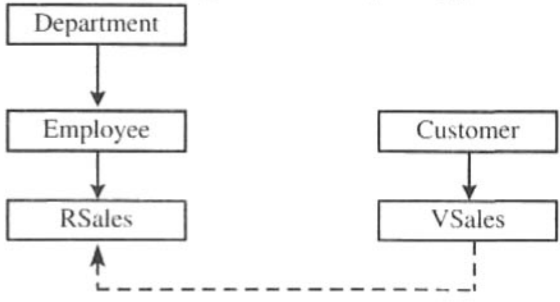


التحويل من نموذج قاعدة البيانات العلاقية إلى نموذج قاعدة البيانات الهرمية : Converting From Relational to Hierarchic Database Model

كما تم التحويل من قاعدة البيانات العلاقية إلى قواعد البيانات الشبكية في حالة علاقة الربط واحد - لمتعدد كذلك ايضاً في قاعدة البيانات الهرمية سوف تظل كما هي. ولكن ملف (المبيعات Sales) الذي يمثل ملف الوصل؛ يوجد سجلان كلاهما أب له؛ لذا فإنه ينبغي أن يوضع في شجرتين. إحداهما يجب أن يلحق بها كملف حقيقي (Rsale) والأخرى يلحق بها كملف افتراضي (Vsale) كما هو مبين في الشكل رقم (٩ -١٨).



شكل رقم (٩-١٨) الهيكل الهرمي لمثال إدارة التخزين



٢- التحويل المنطقى من نماذج قواعد البيانات التبحرية إلى نموذج قاعدة البيانات العلاقية:

التقنية العلاقية أصبحت مقبولة بشكل متزايد فى معالجة البيانات التجارية وأصبح تحول عدد كبير من قواعد البيانات التبحرية إلى قواعد البيانات العلاقية أمراً حتمياً. وإنه من المهم فهم كيفية معرفة إجراء تعديلات للتصميم المادى وتطوراته فى قواعد البيانات التبحرية وكيفية التحويل إلى قواعد البيانات العلاقية المكافئة القابلة للتطبيق[٣].

مثال (٩-١):

يعبر هذا المثال عن بيانات مكتب تعاقدات والموارد المالية من قبل الفنادق والمطاعم وجذب السياحة. والتى تهدف إلى جذب الاتفاقيات للمدينة، ولإتمام ذلك يجب أن تبقى البيانات فى فنادق القصيبى. ولكل فندق سجل به اسم الفندق HName وعنوانه HAddr واسم المدير HMang وعدد الغرف HNumRms . ولكل غرفة فى الفندق يسجل رقم الغرفة RNum ، والسعر اليومى RPrice ، الحجم بالمتر المربع RSize وبالنسبة للأثاث يسجل لكل قطعة الرقم التسلسلى FSer والنوع FType واللون FColor. ولكل هيئة يسجل اسم التنظيم OName، والعنوان OAddr وعدد الأعضاء ONumMems، ويسجل لكل رئيس القسم الاسم XName والعنوان XTitle. ولكل اتفاقية سابقة لأحدى الهيئات سجل يحتفظ بسنة توقيعها CYear وفترتها الزمنية بالأيام CNumDays وعدد الأشخاص الذين وقعوها CAtten. بالإضافة إلى الاحتفاظ بقائمة الفنادق التى لها اتفاقيات خاصة ، وقائمة للاتفاقيات التى استعملت فندقاً خاصاً لمدة أكثر من سنة. وسواء ضمن فى الاتفاقية معلومات أو لم يضمن يعد الفندق صاحب الاتفاقية هو المركز الرئيسى للاتفاقية HQ وعدد غرفه المستعملة من قبل الاتفاقية RmsTaken. وبفرض أن الحقول التى لا تكرر هى اسم الفندق HName واسم الهيئة OName وداخل كل فندق رقم الغرفة RNum ورقم قطعة الأثاث FSer وسنة الاتفاقية السابقة CYear. واسم رئيس القسم داخل التنظيم XName.

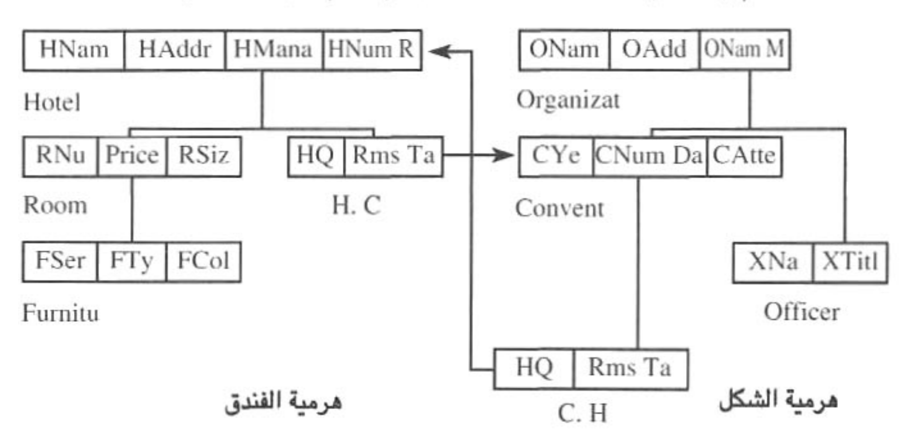
فى نظام إدارة المعلومات IMS :

يبين الشكل رقم (٩-١٩) البيانات مرتبة طبقًا لقاعدة بيانات نظام إدارة المعلومات IMS. وتتكون قاعدة البيانات من شجرتين (هرميتين) هما:

- هرمية الفندق المرتبطة بجزئية الفندق Hotel Segment كأصل root لها.

- هرمية التنظيم المرتبطة بجزئية التنظيم Organization Segment كأصل root لها.
وهاتان الهرميتان (شجرتان) متصلتان بشكل متداخل من خلال علاقات الربط المنطقية ذات الاتجاهات المزدوجة . وفى نظام إدارة المعلومات IMS نجد أن كل جزئية تشير إلى الجزئية التالية لها فى هيكل قاعدة البيانات التبحرية والتى تمثل علاقة ربط واحد - لمتعدد .

الشكل رقم (٩-١٩) قاعدة البيانات الهرمية (IMS) لبيانات مكتب الفنادق

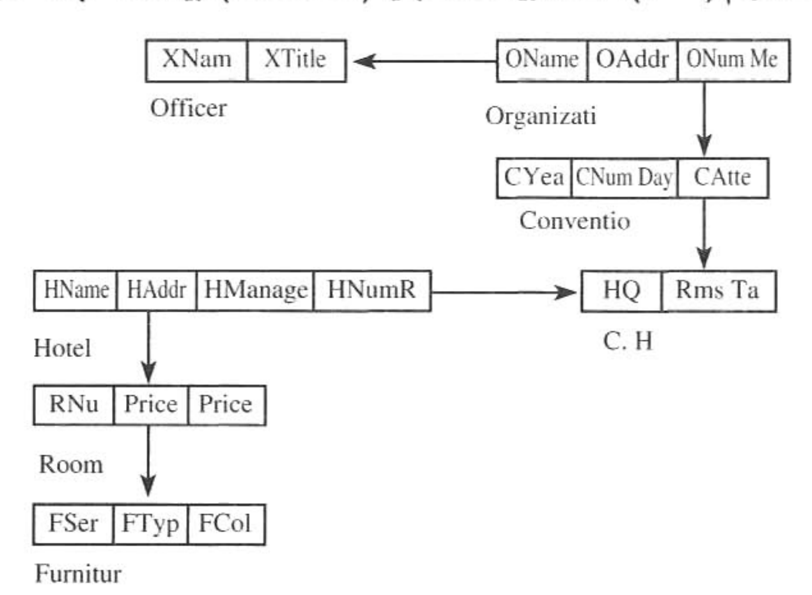


وكما هو واضح من الشكل رقم (٩-١٩) فإن الاتصال المادى بين قيم جزئية الفندق Hotel وقيم جزئية الغرفة Room يشار إليه بقيم الغرف التى تنتمى إلى قيم الفندق وعلاقات الربط متعدد - لمتعدد بين الاتفاقيات Conventions والفندق Hotels تكون مدعمة بالاتجاه المزدوج biderictional، والتى تمثل علاقات الربط المنطقية. ويمثل حقلا المراكز الرئيسية للاتفاقيات HQ وعدد الغرف المستعملة من قبل الاتفاقية RMSTAKEN البيانات المتقاطعة Intersection data لجزئيتى الأبناء المنطقية H-C ، C-H.

فى النظام التشاورى فى لغة نظام البيانات CODASYL:

يبين الشكل رقم (٩-٢٠) قاعدة البيانات الشبكية فى النموذج التشاورى فى لغة نظام البيانات CODASYL ، حيث يتم تمثيل علاقة الربط واحد - لمتعدد بيانات المكتب بسهم على كل نوع سجل Record type . وعلاقة الربط متعدد - لمتعدد بين الفنادق Hotels والاتفاقيات Conventions يتم تحقيقها بسجل الوصل Juncture record المسمى H-C.

الشكل رقم (٩-٢٠) قاعدة البيانات الشبكية (CODASYL) لبيانات مكتب الفنادق



التحويل إلى النموذج العلاقى :

طبقًا لقواعد التحويل المنطقية من النماذج التبحرية إلى النموذج العلاقى فإن قاعدة البيانات لنظام إدارة المعلومات IMS فى الشكل رقم (٩-١٩)، وكذلك قاعدة البيانات للنظام التشاورى فى لغة نظام البيانات CODASYL فى الشكل رقم (٩-٢٠) يتشابهان فى التحويل إلى نموذج قاعدة البيانات العلاقية فى شكل (٩-٢١) الناشئ من الطريقة التالية:

١- جزئيتا الأصل اللتان تمثلان نوعى السجلين الفندق Hotel والتنظيم Organization يتم تحويلهما إلى مخططين علاقيين بدون حقول إضافية.

٢- تحويل جزئية الغرفة Room إلى مخطط علاقى ولكن مع إضافة حقل اسم الفندق HName كمفتاح أجنبى لإنشاء علاقة الربط واحد - لمتعدد بينه وبين جزئية الفندق Hotel؛ ليتم تعريف الغرف بشكل تام، حيث إن رقم الغرفة RmsNum لا يتكرر داخل كل فندق . وكذلك تتشابه جزئية الأثاث Furniture التى يجب أن يتم تحويلها إلى مخطط علاقى يحتوى على خاصيتى اسم الفندق HName ورقم الغرف RNum. وتصبح كل من جزئيتى/نوعى السجلين الاتفاقيات Convention رئيس القسم Officer مخططين علاقيين مستقلين مضافًا إلى كل منهما خاصية اسم التنظيم OName لتعريف الاتفاقية الخاصة بالتنظيم وأن رئيس القسم المعين ينتمى إلى التنظيم المحدد.

٣- وهناك علاقة ربط واحدة فى هذا المثال بين الفنادق Hotels والاتفاقيات Conventions، والتى تم تطبيقها من خلال علاقة الربط المنطقية فى شكل (٩-١٩) باستعمال جزئيتى الابنين C-H ، H-C والتى تكون إحداهما جزئية (ملف) حقيقية والأخرى جزئية (ملف) افتراضية تتضمن المؤشرات، والتى يتم تمثيلهما فى شكل (٩-٢١) بملف الوصل H-C.

٤- وعلاقة الربط هذه يمكن تحويلها بإنشاء مخطط علاقى جديد يسمى التسجيلات bookings. وتتضمن هذه العلاقة خاصيتى اسم الفندق HName واسم التنظيم Oname.

الشكل رقم (٩-٢١) قاعدة البيانات العلاقية لبيانات مكتب الفنادق

Hotel

HName	HAddr	HManager	HNumRms

Room

HName	RNum	Price	RSize

Furniture

HName	RNum	FSer	FType	FColor

Organization

OName	OAddr	ONum Mems

Convention

OName	CYear	CNum Days	CAttend

Officer

HName	OName	XName	XTitle

Bookings

HName	OName	CYear	HQ	Rms Taken

التحويل المادى لقاعدة البيانات Physical Database Conversion :

تصميم قاعدة البيانات المادى هو عملية تعديل أو تطوير نتائج تصميم قاعدة البيانات المنطقى لأغراض الأداء. ولا توجد منهجية شكلية لتصميم قاعدة البيانات المادية بنفس السياق حيث إن تطبيع البيانات هو منهجية تصميم قاعدة البيانات المنطقية . ولابد من أخذ هذه التوصيات فى الحسبان عند التحويل المادى من قاعدة البيانات التبحرية إلى قاعدة بيانات علاقية [٣]:

١- معرفة ما إذا كانت قواعد البيانات التبحرية قد تم تعديلها وكيف تم التعديل لأغراض الأداء فى الوقت الذى صممت فيه.

٢- تحديد عوامل أداء التطبيق الحرجة التى تم أخذها فى الحسبان عند تعديل قاعدة البيانات.

كلا الخطوتين (١) ، (٢) من الأهمية بمكان أن يأخذا بعين الاعتبار ما إذا كانت لا توجد توثيقات كافية وملائمة.

٣- معكوس عملية تعديل الأداء فى الهياكل التبحرية لكى يتم الحصول على نوع الهيكل المبسط الذى يمكن الحصول عليه عند تصميم قاعدة البيانات المنطقية.

٤- تحويل الهيكل المبسط الذى تم الحصول عليه فى الخطوة السابقة إلى قاعدة البيانات العلاقية باستخدام عملية التحويل المنطقى التى تم توصيفها من قبل.

٥- تعديل أو تطوير قاعدة البيانات العلاقية التى تم الحصول عليها فى الخطوة السابقة لتنظيم الأداء فى حدود عوامل أداء التطبيق المأخوذة فى الحسبان فى الخطوة (٢).

كل أنواع التعديلات الهيكلية التى يمكن أن تتم فى هيكل قاعدة البيانات تعتمد على نموذج البيانات وعلى نظام إدارة قواعد البيانات المستعمل.

قرارات أسلوب التداول Access Method Decisions :

الموضوع الأول هو مقارنة قرارات تصميم الأداء التى تهتم بطرق التداول وتخزين السجل . بالإضافة إلى كونه موضوع مهمًا فى خصوصيته إلا أنه ينبغى أن يضع التخزين لبعض قرارات التحويل فى المرتبة الثانية.

فى التصميم المبدئى لنظام إدارة المعلومات IMS الهرمى ، التنظيمان للملف الرئيسى وقرارات أسلوب التداول التى يجب أن تتم هى:

١- ما نوع التداول التى ينبغى أن يوجد لجزئية الأصل خلال مفتاحه الذى لا يتكرر؟ .

٢- وكيف يتم تخزين قيم الجزئية التابعة Subordinate ؟

أساليب تداول نظام إدارة المعلومات IMS هى:

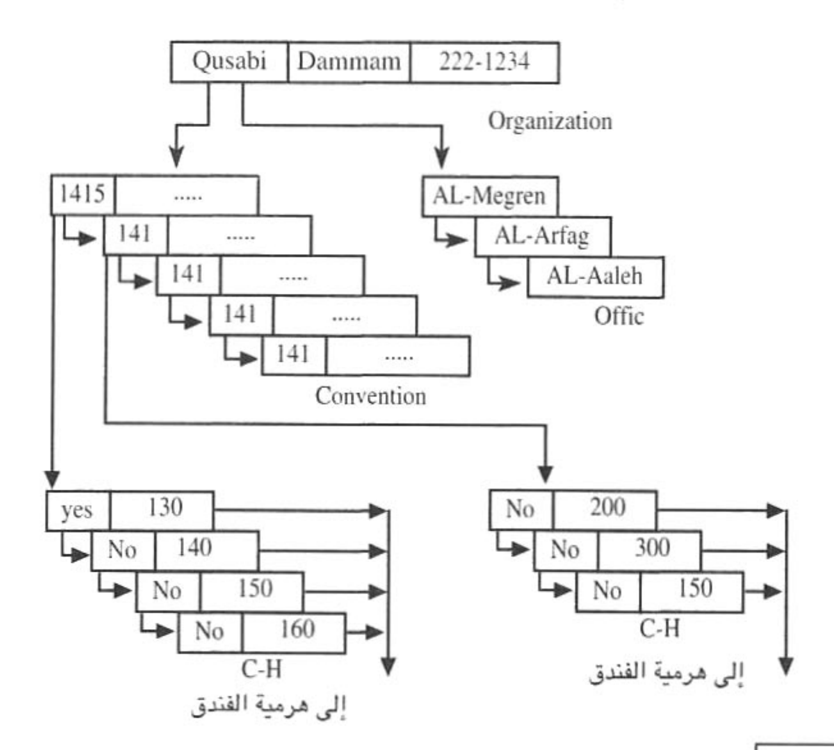
- خيارات تداول سجل الأصل Root هى تسلسلية Sequantial ، فهرسة Indexed ومفرومة hashed.
- خيارات تداول الجزئية التابعة هى تسلسلية وذات مؤشر Pointer based.

التطبيقات الأكثر أهمية ، والعالية السرعة والمباشرة تطبق قواعد بيانات نظم ادارة المعلومات IMS:

١- توفر إما التداول المفهرس أو التداول المفروم لسجل الأصل.

٢- وتوفر التداول المبنى على المؤشر (مؤشر الابن Child والتوائم Twin) للجزئيات التابعة. ويبين شكل رقم (٩-٢٢) ترتيب مؤشرات الابن والتوائم فى الجزء الخاص بهرمية القيم أو سجل قاعدة البيانات.

شكل رقم (٩-٢٢) سجلات قاعدة البيانات الهرمية للتنظيم



إضافة إلى الفهارس الثانوية Secondary Indexes التى قد تنشأ لتوافر التداول المباشر لأى جزئية خلال حقولها (دون الحاجة لأن تكون مفاتيح)، ولكن لايزال يفضل أن يكون مفتاح جزئية الأصل أن يكون مفروماً hashed.

أساليب تداول قواعد البيانات الشبكية هى:

فى النظام التشاورى للغة نظام البيانات CODASYL أنواع السجلات الأعلى والتابعة (التى تعرف بالفئة عند تجميعها) تكون متصلة كل بالآخر بسلاسل دائرية Circular Chains للمؤشرات اللاحقة. وقيم نوع أى سجل معين يمكن استرجاعها خلال قيم السجل الأعلى له عن طريق المؤشرات أو يمكن تخزينها أو استرجاعها عن طريق التفريم hashing إلى جانب سماح بعض التطبيقات بالفهرسة الثانوية.

فى قواعد البيانات التبحرية ، قرارات التصميم المادى التى تهتم بتخزين واسترجاع الجزئية أو السجل تبنى على متطلبات التطبيق التى ينبغى استعمالها. ففى نظام إدارة المعلومات IMS ، التداول المفروم لجزئية الأصل يتم أخذه فى الاعتبار؛ لأنه أسرع من التداول المفهرس، وإن كانت طبيعة التسلسل المفهرس أنه يوفر سرعة نسبية فى فهرس VSAM-KSDS للتداول التسلسلى لجزئية الأصل عبر مفتاحه.

فى قاعدة البيانات الشبكية فى النظام التشاورى للغة نظام البيانات CODASYL يكون التداول المباشر عبر تفريم أى نوع سجل لاختيار ما يكون مغرياً ، ولكن غياب التفريم يوفر تداول أسرع عندما تضطر الفئة الجزئية المرتبة لقيم نوع سجل ما إلى أن تسترجع من نوع السجل الأعلى أو نوع السجل الأب Owner recordtype للفئة التى هم فيها. الفهارس الثانوية فى أى نظام توفر تداولاً مباشراً إضافياً للبيانات، ولكن عيب فيها أيضاً أنها تبطئ النظام أثناء التعديل الصعب حيث النظام يجب أن يتوقف لكى يتم تعديل أى فهرس ثانوى يتأثر بالتعديل.

أساليب تداول قواعد البيانات العلاقية هى:

ليس لديها المؤشرات التى تربط السجلات والاستعمال الصعب للفهارس. بالإضافة إلى أن بعض النظم العلاقية مثل الإنجرس INGRES تسمح بتخزين واسترجاع سجلات جدول ما بالتفريم Hashing.

بعد إنجاز الاشكال المنطقية للتحويل ، فإن أحد المظاهر المادية الأساسية هو التأكيد على السرعة حيث إن التداول المباشر المتوافر فى قواعد البيانات التبحرية يكون متوافراً أيضاً فى قواعد البيانات العلاقية . للبدء بكل جدول علاقى يتم استنتاجه مباشرة من جزئية الأصل فى نظام إدارة المعلومات IMS الهرمى أو السجل الشبكى المفروم فى النظام التشاورى للغة نظام البيانات CODASYL يجب أن يكون مفهرساً أو مفروماً على حقول المفتاح المكافئ. والمثال على ذلك هو حقل اسم الفندق HName فى جزئية الفندق Hotel لهرمية الفندق فى شكل رقم (٩-١٩) وحقل اسم الفندق Name فى شكل رقم (٩-٢١). كل فهرس ثانوى فى قاعدة البيانات التبحرية يجب أن يعاد بناؤه كفهرس فى الجدول العلاقى المناسب. الفهرس الثانوى فى نظام إدارة المعلومات يمكن أن يبنى على حقل فى الجزئية التابعة (المصدر) لكي تسترجع الجزئية الأعلى (المستهدفة). على سبيل المثال الفهرس الثانوى يمكن أن يبنى على حقل الرقم التسلسلى FSER لجزئية الأثاث لهرمية الفندق فى شكل (٩-١٩) لاسترجاع قيم الغرفة للغرف التى توجد بها قطع الأثاث. ومن المهم ملاحظة أن الجدول العلاقى للأثاث فى شكل (٩-٢١) يوجد له من قبل حقل اسم الفندق HName وحقل رقم الغرفة RNum التى تعرف الغرفة التى لها أثاث. لو أن كل المطلوب للفهرس الثانوى هو إيجاد معرف الغرفة room identifier ، عندئذ يكون الجدول العلاقى للأثاث هو فقط الجدول المطلوب فى عملية التداول. لو أن خاصية بيانات أخرى كانت مطلوبة للغرفة Room أو الفندق Hotel، فإن الفهرس الثانوى المكافئ فى نظام إدارة المعلومات IMS الموصف أعلى سوف يكون فهرساً على حقل رقم الأثاث FSER لجدول الأثاث بالإضافة إلى الربط النهائى eventual join لذلك الجدول مع جدول الغرفة Room أو جدول الفندق Hotel أو كليهما.

وأخيراً، ماذا لو كان القرار هو توفير تداول تسلسلى مفهرس (أى أن تداول B-tree) لقيم سجل الأصل لاسترجاع قيم الفئة الجزئية المرتبة بشكل كفء ؟ التكافؤ فى قاعدة البيانات العلاقية هو أن يتم تجميع الجدول العلاقى على حقل مفتاحه . فعلى سبيل المثال، لو كانت جزئية الفندق Hotel لهرم الفندق فى شكل رقم (٩-١٩) كانت مرتبة بطريقة التسلسل المفهرس باستعمال حقل اسم الفندق Hname ، عندئذ الفهرس المجمع Clustered index ينبغى أن يبنى فوق حقل اسم الفندق Hname للجدول العلاقى

للفندق Hotel فى شكل رقم (٩-٢١)، ليس فقط لفهرسة السجلات على ذلك الحقل ولكن أيضًا لترتيب سجلات الجدول طبقًا لقيم ذلك الحقل بشكل طبيعى.

المكان الملائم للجزئية /السجل Segment / Record Placement :

تداول البيانات أو استرجاعها فى النظم التبحرية يعنى التبحر خلال هياكل التخزين الشبكية والهرمية والاسترجاع مرتبط بقيم السجل. وفى حدود الأداء فإن هذه النظم حاسة للمسافة على القرص من البيانات التى تم بحثها للإشارة إلى المدخل (سجل الأصل فى نظام إدارة المعلومات IMS) فى الهيكل. الاسترجاع يكون أكثر كفاءة لو كانت البيانات فى نفس الكتلة Same block أو على الأقل نفس الأسطوانة Same Cylinder كنقطة للمدخل. ويمكن والقيم الجزئية فى نظام إدارة المعلومات IMS داخل سجل قاعدة البيانات أن تخزن فى شكل تسلسلى أو شكل ذى مؤشر (فى حالة أغراض الأداء ذات التداول المباشر). فى حالة أخرى ، التخزين المادى لقيم الجزئية سوف يكون فى شكل قوائم تسلسلية مسبقة الترتيب:

من أعلى إلى أسفل .Top-to-bottom :

من اليسار إلى اليمين Left-to-right .

ومن الأمام إلى الخلف Front-to-back .

وحالة الترتيب من الأمام إلى الخلف تعنى التحرك خلال قيم سلاسل التوائم لنوع الجزئية (الشجرة) Segment type موضع الانتقال . فى هرمية التنظيم Organization hierarchy . هذا يعنى أن كل قيم جزئية الاتفاقية وجزئية الابن المنطقى C-H لتنظيم خاص ينبغى أن يتم تخزينها بعد قيمة الأصل root accurrence وقبل قيم رئيس القسم officer. فى حين أن القاعدة العامه فى مثل هذه الحالة هو فرض أن قيم رئيس القسم ينبغى ألا تقطن فى نفس المنطقة (مساحة) القابلة للعنوانة بشكل مادى على القرص مثل قيم التنظيم Organization accurrences بسبب كل التحويلات وقيم الابن المنطقى C – H التى يجب أن تكون مخزنة فيما بينهم.

قاعدة إصبع الإبهام The rule-of -thumb :

فى هرميات نظام إدارة المعلومات IMS فإن قاعدة إصبع الإبهام تجهز لوضع الجزيئات التى يتم تداولها بشكل متكرر فى أقصى علو high up ويساراً بقدر الإمكان فى الهرم (شجرة). على سبيل المثال: هرمية التنظيم Organization hierarchy فى شكل رقم (٩-١٩) لها جزئية الاتفاقية إلى اليسار من جزئية رئيس القسم. مناظرة التصميم المنطقى بالمادى فى عملية التصميم المنطقى أيضًا قد يكون لها انعكاس. ومن وجهة نظر الأداء، لو أن بيانات الاتفاقية قد تم تداولها بشكل أكثر تكراراً من بيانات رئيس القسم ، فإن التصميم المادى يصر على أن جزئية الاتفاقية تكون على اليسار من جزئية رئيس القسم. والوضع الذى يمكن أن يأخذ فى االاعتبارهو أن الحالة العملية المكافئة لذلك الإصدار فى الحالة العلاقية ليست مظهراً أساسيًا للنموذج العلاقى. وأكثر من ذلك، فإنه من المهم تجميع جداول علاقية مترابطة معًا فى مساحة تخزين عريضة لغرض زيادة كفاءة عمليات الربط (على سبيل المثال مساحات جداول DB2) .

مشكلة أخرى ممكنة فى قاعدة بيانات نظام إدارة المعلومات IMS هى فى ملكية نوع الجزئية التابعة التى يكون لها قيمة واحدة بالنسبة إلى قيمة جزئية الأب الخاص بها. مثال ذلك لو أنه فى هرمية التنظيم فى الشكل رقم (٩-١٩) حلت جزئية رئيس President segment محل جزئية رئيس القسم officer segment حيث يوجد فقط رئيس واحد له قيمة فى جزئية التنظيم. وأكثر من ذلك وجود مساحة لجزئية الرئيس وبياناته يمكن دمجها فى جزئية التنظيم التى لم تفقد البيانات سلامتها.

الوضع المكافئ فى قاعدة البيانات العلاقية ينبغى أن يتكون من جدولين علاقيين مترابطين بعلاقة ربط واحد - لواحد، أحدهما له مفتاح خارجى يستخدم لربطه بالجدول الآخر. ونتاج هذا الربط يكون بشكل دائم مثل الحالة التى فى نظام إدارة المعلومات IMS.

فى التصميم المادى لنظام إدارة المعلومات IMS تأخذ نتيجة عملية التصميم المنطقى وتنتج عدداً من أنواع الجزئيات بتجميع أنواع الجزئيات معًا أو بزيادة عددها بتقسيم بعضها (أو كلها) إلى أنواع جزيئات أخرى. ونفس الإجراء يمكن أن يتم فى

أنواع السجلات الشبكية فى النظام التشاورى للغة نظام البيانات CODASYL. وتجميع جزيئات الأب والابن فى نظام إدارة المعلومات IMS معًا قد يقلل مقدار التبحر الذى قد يحتاجه لاسترجاع البيانات ولكن يؤدى إلى زيادة البيانات عن الحد. تقسيم الجزيئات قد يتم لتحسين الأمن واستقلالية البيانات ولكنه قد ينتج هيكلاً معقداً.

فى التحويل إلى قاعدة البيانات العلاقية ، يمكن أن يتم تعريف تقسيم الجزيئات فى نظام إدارة المعلومات IMS بوجود جزئيتين بنفس حقل المفتاح فى نفس المستوى فى الهرم (الشجرة). مثال ذلك تقسيم جزئية الغرفة Room Segment لهرمية الفندق إلى جزئيتين سوف ينتج جزئية. تتكون من حقلى رقم الغرفة RNUM وسعر الغرفة RPRICE وجزئية أخرى تتكون من حقلى رقم الغرفة RNum وحجم الغرفة RSize. وهذه يمكن أن يعاد تجميعها عند التحويل لقاعدة البيانات العلاقية أو تحول كما هى مع إضافة حقل اسم الفندق HName لكل جدول علاقى ناتج كمفتاح خارجى.

بتجميع الجزيئات فى نظام إدارة المعلومات IMS يمكن أن يتم تعريفه بواسطة تنفيذ تحليل التطبيع Normalization analysis لحقوله أو بملاحظة البيانات الفعلية المخزنة الزائدة عن الحد redundancy . ويجب ملاحظة أهمية عدم التطبيع Denormal-ization الذى يعنى تجميع الجداول العلاقية المترابطة لقاعدة البيانات العلاقية معًا: لتقليل عدد عمليات الربط الضرورية فى الاستفسار. وهذا يؤدى أيضًا إلى زيادة البيانات عن الحد ولكن من وجهه نظر التصميم المادى. ويلاحظ أن ما يناظر عدم التطبيع فى قاعدة البيانات العلاقية هو نفس تجميع الجزيئات المتصلة معًا لقاعدة البيانات الهرمية فى نظام إدارة المعلومات IMS أو لطى Collapsing الفئات فى السجلات لقاعدة البيانات الشبكية فى النظام التشاورى للغة نظام البيانات CO-DASYL . ويوجد مؤشرات اختيارية متنوعة يمكن أن يقدمها نظام إدارة المعلومات IMS أو النظام التشاورى للغة نظام البيانات لتحسين الأداء. مثال ذلك: مؤشرات توائم الاتجاه العكسى Twin backword pointers فى قواعد بيانات نظام إدارة المعلومات IMS ومؤشرات الأب Owner Pointers فى قواعد البيانات الشبكية فى النظام التشاورى للغة نظام البيانات CODASYL. ولو كان الغرض من المؤشر الاختيارى هو لسرعة عملية نظام إدارة قاعدة البيانات DBMS دون الحصول على استعمال التطبيق

المنطقى (كمؤشرات توءم الاتجاه العكسى فى نظام إدارة المعلومات IMS لتحسين سرعة الحذف على سلاسل التوءم)، عندئذ لا يوجد نموذج علاقى مكافئ لذلك. ومن ناحية أخرى، على سبيل المثال فى النظام التشاورى للغة نظام البيانات CODASYL فإن مؤشر الأب يوفر أسرع أسلوب لإيجاد قيم السجل الأعلى. والمكافئ له فى قاعدة البيانات العلاقية هو أن معرف كينونة السجل الأعلى هو من قبل فى القيم المرتبة كمفتاح خارجي بوصفه ناتجًا لعملية التصميم المنطقية. لو كان الاحتياج لأكثر من قيمة المفتاح فإن الربط يجب أن ينفذ مع الجدول العلاقى الذى يمثل كينونة السجل الأعلى.

عمليات التخزين والاسترجاع المباشر Direct Storage-and -retriverd Operations :

تخزين واسترجا ع السجلات المبنى على حسابات قيم المفاتيح هو ما يسمى بالتفريم، التفريم يعتبر الأسرع بشكل عام لتنفيذ عمليات التخزين والاسترجا ع المباشر. ويسمح نظام إدارة المعلومات IMS بالتداول المباشر لجزئية الأصل root Segment باستعمال خيار التفريم hashing أو الفهرسة index. الجزيئات الأخرى بخلاف الأصل root يمكن الوصول إليها مباشرة باستعمال الفهارس الثانوية Secondary Indexes وليس بالتفريم hashing.

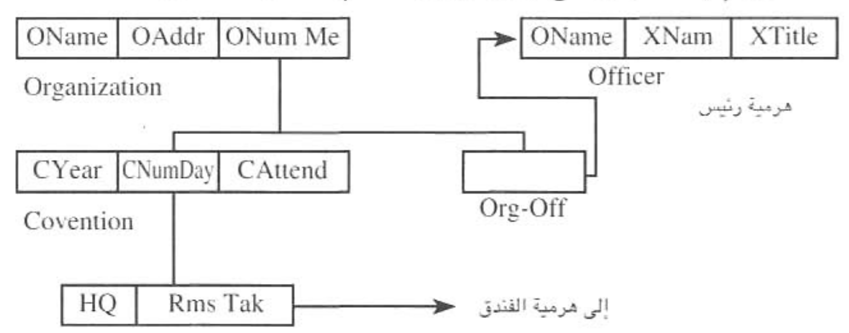
فى تصميم هرمية التنظيم فى شكل رقم (٩-١٩) فإنه يبدو متوازنًا بشكل تام فى حدود التصميم المنطقى ، ويبين شكل رقم (٩-٢٢) أيضًا جزءًا لقيم هرمية التنظيم لترابطات Qusabi . بافتراض أن تطبيقًا حرجًا يتطلب سرعة وتداولاً مباشراً لجزئيات رئيس القسم Officer Segments ؟ يوجد طرق قليلة يمكن من خلالها الوصول إلى رئيس القسم المعين على أساس مباشر. حيث إن اسم رئيس القسم لا يتكرر داخل التنظيم . فإن النظام يمكن أن يجد قيمة التنظيم مباشرة بالفهرس أو التفريم اعتماداً على البيانات المخزنة فى أحدهما وبعدها يتم التبحر عبر المؤشرات إلى قيم رئيس القسم التى يتم بحثها. وهذا يتطلب عملية إدخال وإخراج إضافية أو أكثر من عملية لهيكل هرمى أكثر تعقيداً.

هناك بديل آخر لتداول رئيس القسم مباشرة ويتم بالحصول على فهرس ثانوى يبنى فوق حقل اسم رئيس القسم XName لجزئية رئيس القسم لتداوله مباشرة. ولكن

ماذا يحدث لو كان هناك أكثر من تنظيم له نفس اسم رئيس القسم؟ فى هذه الحالة ستكون عمليات إدخال وإخراج اضافية سوف تجلب نوعًا آخر للمؤشر التالى (مؤشر الأب) للرجوع إلى قيم الأصل root occurrences لتميز التنظيم الصحيح. بديلاً آخر هو نسخ حقل اسم التنظيم Oname لإضافته إلى جزئية رئيس القسم وبناء الفهرس الثانوى على حقلى اسم التنظيم Oname واسم رئيس القسم Xname ينشئ شكلاً من البيانات الزائدة عن الحد. وهذا لايزال بعيداً اعتماداً على الفهرس للتداول المباشر كمقاوم لإجراء التفريم الأسرع.

ويبين شكل (٩-٢٣) الحل البديل لمشكلة أداء نظام إدارة المعلومات .IMS وقيم جزئية رئيس القسم لازال يمكن الوصول إليها خلال قيم جزئية تنظيم Orgamization عبر علاقة ربط موحدة الاتجاه Unidirection لربطهم وهو ما يطلق عليه الجزئية الافتراضية Virtual segment. ومزية ذلك هى أن أصل الهرم (قيم جزئية رئيس القسم) تستعمل حقلى اسم الرئيس القسم Xname واسم التنظيم Oname معًا كمفتاح يمكن أن يخزن ويسترجع عبر إجراء التفريم، وهذا سوف يوفر السرعة للتداول المباشر بقدر الإمكان . ويلاحظ فى قواعد البيانات الشبكية للنظام التشاورى للغة نظام البيانات CODASYL أن أى نوع سجل يمكن أن يتم تخزينه بإجراء التفريم.

شكل رقم (٩-٢٣) يوضح جزئية رئيس القسم كأصل لقاعدة بيانات منفصلة



وقرار تحديد أنواع سجلات يتم تفريمها أيًا كانت وكذلك أى سجلات يتم التخزين فيها فقط بالمؤشرات يؤدى إلى التساؤل عن كيفية حصول قيم مترابطة بشكل مادى منتشرة على القرص؟ وهكذا فى النظام التشاورى للغة نظام البيانات CODASYL فى

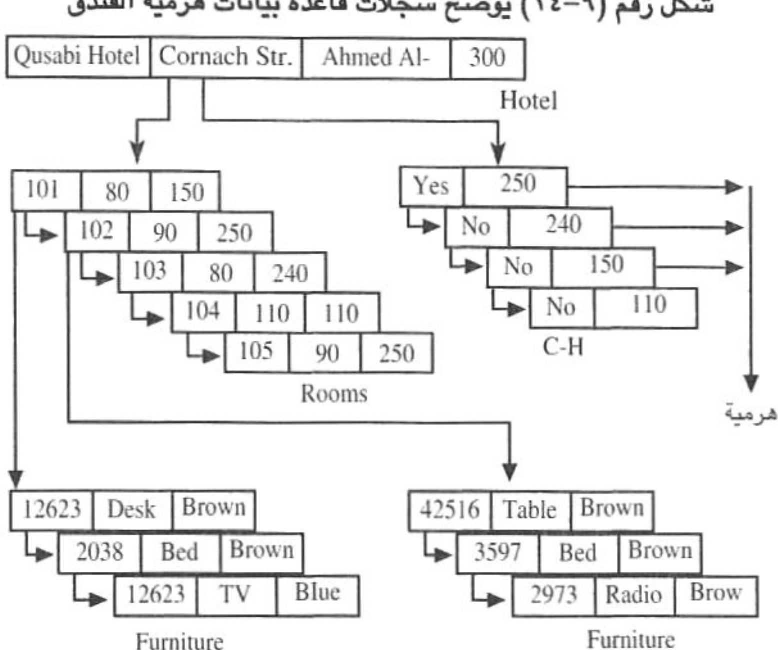
شكل رقم (٩-٢٠)، نسخة من حقل اسم التنظيم OName مضافة إلى سجل رئيس القسم Officer الذى يمكن من التخزين عندئذ عبر التفريم باستخدام حقلى اسم رئيس القسم XName واسم التنظيم OName كمفتاح بنفس الطريقة المستخدمة فى نظام ادارة المعلومات IMS. ولكن ماذا يحدث لو كان القرار الأصلى خاصًا بتعديل هرم التنظيم فى شكل رقم (٩-١٩) إلى هرميات شكل (٩-٢٣) لاسباب تتعلق بالأداء ولتحويلة إلى قاعدة بيانات علاقية؟ والإجابة تكمن فى حالة عدم وجود توثيق صريح، فان الخطوة الأولى هى معرفة الهياكل التى تم تصميمها. فصل قاعدة البيانات، علاقة الربط المنطقية ذات الاتجاه الموحد وتكرار حقل اسم التنظيم Oname هى مفاتيح حل هذا اللغز. النتيجة هى معرفة علاقة الربط واحد - لمتعدد بين التنظيمات Organizations ورؤساء الأقسام Officers ومن وجهة النظر المنطقية لنظام إدارة المعلومات IMS أو للنظام التشاورى للغة كنظام البيانات CODASYL، وتكرار حقل اسم التنظيم ONAME فى جزئية رئيس القسم Officer. بهذا يكون التحويل المباشر الذى ينتج فى جداول التنظيم Organization ، رئيس القسم Officer، الاتفاقية Convention والتسجيلات bookings فى شكل رقم (٩-٢١).

إن إضافة حقل اسم التنظيم OName فى جزئية رئيس القسم فى شكل رقم (٩-٢٣) التى تمت لأسباب تتعلق بالأداء، - ضرورة منطقية فى قاعدة البيانات العلاقية فى شكل (٩-٢٠) حيث إن حقل اسم التنظيم OName يتم وضعه فى الجدول العلاقى الخاص برئيس القسم Officer كمفتاح خارجى. وللحصول على السرعة والتداول المباشر لرؤساء الاقسام Officers فى قواعد البيانات العلاقية، فإن حقلى اسم التنظيم OName ورئيس القسم XName فى الجدول العلاقى لرئيس القسم يمكن أن يستعملا كمفتاح للفهرس (كما فى دى. بى ٢ DB2) أو كمفتاح لإجراء التفريم (كما فى لغة الإنجرس (INGRES). فى الشكل رقم (٩-١٩) ترابط سجلات التنظيم وسجلات رئيس القسم فى قاعدة البيانات العلاقية ليس محتملاً أن تكون قريبة من بعضها البعض على القرص لو لم يعرض النظام العلاقى ماديًا تداخل قيم السجلات من مختلف الجداول . وهذا له بعض التأثير على أداء عملية الربط تمامًا كما فى الانتشار المادى للرسم التخطيطى الهرمى فى شكل (٩-٢٢)؛ مما يؤثر على سرعة التبحر خلال الهرم.

سلاسل التوءم الطويلة Long Twin Chains :

ترتبط مشكلة سلاسل التوءم الطويلة بموضوع قوة التفريع حيث يبين شكل (٦) جزءاً من قيم هرم الفندق . وسوف يوجد عدد ضخم من الغرف فى الفندق ولذلك عدد ضخم من قيم جزئية الغرفة Room Segment تلحق بقيمة جزئية واحدة للفندق[٣]،[٤].

شكل رقم (٩-٢٤) يوضح سجلات قاعدة بيانات هرمية الفندق



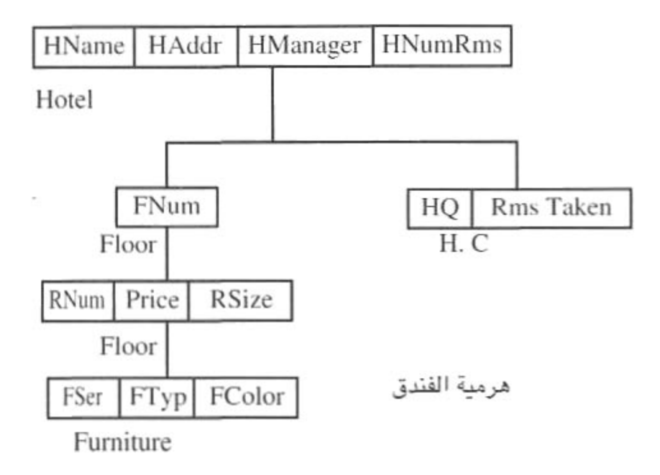
كما تبين جزئية الأصل فى الشكل السابق أن فندق القصيبى Quosabi hotel له (٣٠٠) غرفة؛ ولذلك يوجد (٣٠٠) قيمة لجزئية الغرفة متصلة بواسطة مؤشرات التوءم فى سلاسل التوءم . وقيمة كل غرفة يمكن أن ترتبط بقيم عديدة لجزئية الأثاث Furniture Segment التى تجعل الهيكل الهرمى ككل أكثر ضخامة وانتشاراً وتعقيداً.

إن التطبيق الذى يطلب تداول كل الغرف فى فندق معين يمكن أن يتبع بكفاءة سلاسل توءم الغرفة لكى يتم استرجاع كل قيم غرف الفندق. ولكن ماذا عن التطبيق الذى يتطلب تداولاً مباشراً لغرفة واحدة فى فندق معين؟ التطبيق يمكن أن يصل إلى

قيمة أصل الفندق بسرعة كافية، ولكن عندئذ يجب أن يتم الحرث خلال سلسلة التوءم ، فى الشكل التسلسلى حتى يتم إيجاد الغرفة التى يتم البحث عنها. نفس الوضع سوف يكون متشابهًا فى قاعدة البيانات الشبكية للنظام التشاورى للغة نظام البيانات -CO DASYL كما فى شكل رقم (٩-٢٠)، لو أن نوع سجل الغرفة مخزن عبر الفئة، نسبة إلى نوع سجل الفندق Hotel ، أكثر من كونه مخزنًا عبر حساب إجراء التفريم. ويمكن تطبيق الفهرس الثانوى لجزئية الغرفة، ولكن لأن العديد من الفنادق لها نفس أرقام الغرف، فإن جزئية الفندق سوف يتم تداولها أيضًا مع السعر Price. حقل اسم الفندق Hname يمكن أن يضاف إلى جزئية الغرفة والفهرس الثانوى إلى السعر المكرر. وفى أسلوب أخر فإن جزئية الغرفة يمكن أن ترقى وتصبح أصلاً بنفس الأسلوب المتشابه للهيكل فى شكل رقم (٩-٢٣). التقنية الثالثة هى لإنشاء جزئية ذات مستوى مباشر Intermediate-Level Segment بين الجزئية مع سلاسل التوءم الطويلة "الغرفة" ROOM وجزئية الأب الخاص بها "الفندق" HOTEL. وشكل رقم (٩-٢٥) يبين هرمية الفندق مع نوع الجزئية الجديدة "الطابق" FLOOR المدرجة بين الفندق Hotel والغرفة Room. وبهذا الهيكل يمكن للتطبيق التحرك من الفندق إلى الطابق إلى الغرف على ذلك الطابق دون أن يصادف الغرف على الطوابق الأخرى.

فى التحويل من قاعدة البيانات الهرمية لنظام إدارة المعلومات IMS إلى قاعدة البيانات العلاقية ، فإن التعديل الهيكلى المبين فى شكل (٩-٢٥) يجب أن يكون معروفًا وسبق التعامل معه . جزئية الطابق Floor Segement يجب أن تحذف قبل التحويل إلى قاعدة البيانات العلاقية. بعد التحويل إلى قاعدة البيانات العلاقية ، الجدول العلاقى للغرفة المبين فى شكل رقم (٩-٢١) يجب أن يشمل حقل اسم الفندق HName كمفتاح خارجى. وبناء على إمكانيات النظم العلاقية فإن الجدول العلاقى للغرفة يمكن أن يكون مفرومًا أو مفهرسًا على حقلى اسم الفندق HName ورقم الغرفة RNum لزيادة السرعة والتداول المباشر لغرفة معينة.

شكل رقم (٩-٢٥) هرمية الفندق بعد إضافة جزئية الدور Floor



علاقات الربط متعدد - لمتعدد للنماذج التقليدية

: Many - to - Many Relationships

نوع الهيكل المصمم لتناول علاقات الربط متعدد - لمتعدد فى قواعد البيانات الهرمية لنظام إدارة المعلومات IMS يكون معقداً إلى حد ما. فى حين أن علاقات الربط متعدد - لمتعدد يتم تناولها فى قواعد البيانات الشبكية للنظام التشاورى للغة نظام البيانات CODASYL بأسلوب متشابه بشكل أساسى لنظام إدارة المعلومات IMS، أن يتجه إلى أن يكون أبسط إلى حد ما فى مرحلة التصميم ولكن معقداً لمبرمج التطبيق . فى قواعد البيانات العلاقية كل علاقة ربط متعدد - لمتعدد تتطلب إنشاء جدول إضافى بشكل مباشر.

وفى المثال الحالى: تكون علاقة الربط متعدد - لمتعدد بين الاتفاقيات Conventional والفنادق hotels . كل اتفاقية تستعمل العديد من الفنادق، وكل فندق يستعمل على مدار العام بواسطة عدد اتفاقيات مختلفة. وهذا يوضح بالشكل رقم (٩-١٩) فى قواعد بيانات نظام إدارة المعلومات IMS بواسطة الوصلات بين جزئية الفندق (فى هرمية الفندق) وجزئية الاتفاقية (فى هرمية التنظيم) باستعمال جزيئات الابن المنطقى

C-H C– H .. وفى قواعد البيانات الشبكية للنظام التشاورى للغة نظام البيانات CO-DASYL شكل رقم (٩–٢٠)، عن طريق سجل الوصل Juncture record بين سجلات الفندق والاتفاقيات حيث تعمل H-C لنفس الغرض. ويلاحظ أن قيم جزئية الاتفاقية تشير إلى جزئية الأب الخاص بها؛ لذلك فإن كلتا الجزئيتين يمكن تعريفها لسنة معينة. والهيكل المكافئ فى قواعد البيانات العلاقية فى شكل (٩–٢١) هو جدول التسجيلات bookings الذى يرتبط مع جدولى الفندق والاتفاقية. وهو يتكون من معرفات identifiers كاملة للفندق (Hname) والاتفاقيات OName, CYear بالإضافة إلى البيانات المتقاطعة (RmsTaken , HQ).

ومن القرارات فى نظام إدارة المعلومات IMS علاقة الربط المنطقية وهى إما أن تكون زوجًا ماديًا أو زوجًا افتراضيًا. فى **الزوج المادى**: جزيئات الابن المنطقى مثل C-H , H-C فى شكل رقم (٩–١٩) كلاهما صحيح، بمعنى أن جزيئات مادية تحتوى على مؤشرات وبيانات متقاطعة (أى أن النظام يحتفظ بالمتكرر على كلا الجانبين لعلاقة الربط). وفى **الزوج الافتراضى**: الابن المنطقى الحقيقى يوجد ماديًا على أحد الجوانب لعلاقة الربط فى حين أنه على الجانب الآخر من علاقة الربط يحتوى على الابن المنطقى الافتراضى.

ويوجد العديد من مزايا وعيوب الأداء بين الزوج المادى والافتراضى. وبوجه عام ، لو وجد مقدار كبير من نشاط تداول مهم بشكل متساوٍ لكلا جانبى علاقة الربط المنطقية، فإن الزوج المادى يكون أفضل. فى حالة وجود معظم النشاط على جانب واحد، فإن الزوج الافتراضى يفضل مع وجود الابن الحقيقى فى الجانب الأكثر نشاطًا أو مع الجانب ذى سلاسل التوءم الأطول.

وفى التحويل لقاعدة البيانات العلاقية، نجد أن كل أشكال الهياكل التبحرية لعلاقات الربط متعدد – لمتعدد والتى تشمل الأشكال الهرمية (Virtual file) لنظام إدارة المعلومات IMS والأشكال (Juncture file) الشبكية للنظام التشاورى للغة نظام البيانات CODASYL – سوف تحول إلى جدول علاقى (bookings) كما فى شكل رقم (٩–٢١).

المؤشرات المباشرة والرمزية Direct & Symbalic Pointers :

فى نظام إدارة المعلومات IMS المؤشر من الابن المنطقى إلى الأب المنطقى الخاص به يمكن أن تكون رمزية (قيمة مفتاح متلاصق كلياً للجزئية يشار إليها مثل حقل سنة الاتفاقية Cyear وحقل اسم التنظيم ONAME) ومباشرة (العنوان المادى للجزئية التى يشار إليها) أو كليهما.

المؤشرات المباشرة تفيد فى حد علاقة الربط المنطقية فى الاستفسار بشكل سريع. والمؤشرات الرمزية تفيد فى إعادة التنظيمات وفى مواقع هذه الاستفسارات حيث تكون قيمة مفتاح الأب المنطقى جزءاً فقط من الأب المنطقى الذى نحتاجه. الحالة المكافئة لذلك فى قاعدة البيانات العلاقية هى أن ما ينبغى أن يكون مؤشراً رمزياً فى الابن المنطقى لنظام إدارة المعلومات IMS هو مفتاح كينونة أخرى للجدول العلاقى متعدد-لمتعدد، تجميع حقلى سنة الاتفاقية CYear واسم التنظيم OName فى الجدول العلاقى bookings فى شكل رقم (٩-٢١). وهذا يعنى أن الحالة العلاقية دائماً تشبه المؤشر الرمزى فى حالة نظام إدارة المعلومات IMS. لو كان اسم التنظيم OName وسنة الاتفاقية CYear مطلوبين فقط لتحقيق شرط الاستفسار عند البحث عن كل الاتفاقيات التى تشمل ذلك الفندق، عندئذ يطلب الجدول العلاقى bookings فقط. أما فى حالة طلب معلومات أكثر عن الاتفاقية أو التنظيم عندئذ يجب تنفيذ الربط. إن عملية الربط وظيفياً تكون مكافئة لعبور حد علاقة الربط المنطقية فى نظام إدارة المعلومات IMS سواء أكانت المؤشرات رمزية أم مباشرة.

الهوامش :

1- [RIAD, 1993], Mokhtar Boshra Riad and Halim Habib, "A System of Conversion" between Hierarchic, Network, and Relational Database Models', **The Egyptian Computer Science Journal**, July 1993.

2- [ELMASRI, 1989], Ramez Elmasri, and Shamkant B. Navathe, **Fundamentals of Database Systems**, Benjamin/Cummings, Redwood City, Calif., 1989.

3- [GILLENSON, 1990], Mark L. Gillenson, 'Physical Design Equivalencies in Database Conversion', **Communications of the ACM**, Vol. 33, Number 8, August 1990.

4- [DATE,1990], C.J. Date, **An Introduction to Database Systems**, Volume I, Fifth Edition, Addison-Wesley Publishing Company Inc.,1990.

الفصل العاشر

الاتجاه نحو قواعد البيانات الشيئية الموجهة والتحويل إلى قواعد البيانات العلاقية



مقدمة :

هناك ضرورة حتمية تفرضها أليات سوق قواعد البيانات للاتجاه نحو قواعد البيانات الشيئية الموجهة والتى سبق التطرق إليها؛ لذا من الضرورى التعرف على الأدوات الخاصة بتحليل وتصميم الشىء الموجه واختيار التقنية الملائمة لتصميم النظام. بالإضافة إلى ذلك سيتم استعراض قواعد التحويل من النموذج الشيئى الموجه إلى النموذج العلاقى؛ نظراً للانتشار الواسع الذى تحظى به قواعد البيانات العلاقية حالياً. وسوف يتم التطرق للموضوعات التالية فى هذا الفصل:

منهجيات تحليل وتصميم الشىء الموجه :

سيتم استعراض العديد من النقاط التى تساعد فى عملية اختيار أداة تحليل النظام، إلى جانب سرد طرق ومجالات تطبيقات أدوات تحليل وتصميم الشىء الموجه.

تقنية نمذجة الشىء OMT :

تعتبر هذه التقنية إحدى أقدم المنهجيات فى تحليل وتصميم الشىء الموجه. وتتلخص خطوات الأسلوب المنهجى لهذه التقنية فى:

- تحليل نظام التطبيق.
- تصميم النظام.

التحويل من النموذج الشيئى الموجه إلى النموذج العلاقى :

يتم استعراض قواعد التحويل من النموذج الشيئى الموجه إلى النموذج العلاقى باتباع الخطوات التالية:

- تحويل الأنواع إلى جداول علاقية.
- تحويل التعميم والتخصيص إلى جداول علاقية.
- تحويل التجميعات إلى جداول علاقية.
- تحويل الارتباطات إلى جداول علاقية.

مقارنة بين نماذج البيانات الشيئية الموجهة والعلاقية:

يتم عمل المقارنة بين نماذج البيانات الشيئية الموجهة والعلاقية فى المواضيع المهمة التالية:

- أنوا ع البيانات.
- سلامة البيانات.
- تطوير المخطط.
- معالجة البيانات.

تحليل وتصميم أداة قواعد البيانات الشيئية الموجهة:

تعد أداه تصميم قواعد البيانات الشيئية الموجهة نظامًا برمجيًا يمكن تصميمه لكى يجعل نشاط تصميم قواعد البيانات الشيئية الموجهة أتوماتيكيًا، وذلك للاستفادة منه فى تحويل قواعد البيانات بشتى أنواعها إلى قواعد البيانات الشيئية الموجهة. وسيتم عمل تحليل وتصميم لهذه الأداة باستخدام تقنية الشىء الموجه OMT عن طريق:

- عملية تصميم قواعد البيانات الشيئية الموجهة والتى تتضمن كلاً من عمليات النماذج وخطوات دورة حياة عملية تصميم الأداة.
- تحليل أداة تصميم قواعد البيانات الشيئية الموجهة عن طريق النموذج الشيئى، النموذج المتحرك، والنموذج الوظيفى.

تصميم النظام:

يتم تركيب أداة تصميم قاعدة البيانات الشيئية الموجهة باستعمال البناء المعمارى لنظام أداة التصميم، والذى عن طريقه يمكن للمستخدم إنشاء الأنوا ع، وأنوا ع الوصلات المختلفة والوقائع.

منهجيات تحليل وتصميم الشيء الموجه:

تستطيع أدوات تحليل وتصميم الشيء الموجه معنوياً في تعجيل تقدم هندسة البرمجيات ولكن مطوري النظم والتطبيقات غالباً ما يجدون صعوبة في الاختيار من المنظومة السريعة للمنتجات التجارية. وهناك العديد من النقاط التي ينبغي أن تأخذ في الحسبان عملية الاختيار لأداة التحليل والتصميم، وهي:

١- فائدتها في البحث الطويل الأمد وعدم ثبات السرعة.

٢- تعزيزها لتحسينات هندسة العملية التي تجرى.

٣- تعريفها للبرامج Methods التي تناسب العمليات والتطبيقات موضع التنفيذ.

٤- تخطيطها المتكامل مع البيئة التي يتم من خلالها الاستخدام.

٥- تحديدها لكيفية استعمال النموذج الناتج.

٦- لتوصيفها فقط لمعالم الاحتياجات.

٧- إعطاؤها الأمد الأطول لتسليمها من الإدارة للمورد.

٨- تداولها للفضليات مع الاستعمال التجريبي الحقيقي.

٩- توصيفها لدعم التدريب والاستفسار.

١٠- مدى تفاعلها في الوقت الراهن.

ويشمل الجدول التالي بعض أدوات تحليل وتصميم الشيء الموجه:

طرق ومجالات تطبيقات أدوات تحليل وتصميم الشىء الموجه

الطريقة Method	المؤلفون Author (S)	التطبيقات Applicatns	طريقة النمذجة Modeling Approach
هندسة معلومات الشىء الموجه *Object - Oriented Information Engineering (IE)*	جيمس أوديل - جيمس مارتن	نظم معلومات المنشأه Enterprise information	المخطط : الشىء (الهيكل)، الحدث، الحالة (السلوك) وتدوين هندسة المعلومات IE
تحليل الشىء الموجه *OO Analysis* تصميم الشىء الموجه *OO Design*	بيتركود - إديوردن	نظم المعلومات Information Systems Telecommunication	الطبقات : الموضوع، النوع، الشىء، الهيكل، الخصائص، الخدمات.
هندسة برمجيات الشىء الموجه *Object-Oriented Software Emgimeering*	إيفار جاكسون	نظم معلومات، الوقت الحقيقى Information Systems, Real-Time	تطوير استعمال حالة الاستنتاج، الكينونة، المراقبة وأشياء واجهة التطبيق.
تقنية نمذجة الشىء *Object Modeling Technique*	جيمس رامبو	نظم معلومات، الوقت الحقيقى Information Systems, Real-Time	النماذج الثلاثة : الشيئى، الوظائفى، المتحرك.
تحليل نظم الشىء الموجه الذى يعيد تعريف التصميم *OO Systems Analysis Recursive Design*	سالى شلير	الوقت الحقيقى المضمن معالجة المعاملات *Real-Time embedded transacting Processing*	دورة حياة الشىء والاتصالات : نماذج السلوك ومحاكاتها.
طريقة بوش ٩٣ : تصميم الشىء الموجه	جرادى بوش	نظم المعلومات، الوقت الحقيقى ، الدفاع (الحماية) *Information System, Real-Time, defense*	تصميم *Ada* ، نوع وشىء *C++* ، الحالة والتوقيت
تصميم نظام بلغة الأداء *System design With Ada*	رايموند بهر	*Defense, Rea;-time*	الرسم التخطيطى بمنشئات لغة *Ada*

تعتبر منهجيات الجدول السابق أدوات هندسة برمجيات مساعدة للحاسب. (Computer-Aided Software Engineering CASE) يمكن تصنيفها إلى تصنيفين، التصنيف الأول: الأدوات العليا لهندسة البرمجيات المساعدة للحاسب Upper CASE، والتصنيف الثاني: الأدوات المتكاملة لهندسة البرمجيات المساعدة للحاسب Integrated CAS[١]. كل من هذين التصنيفين يتكون من أدوات أعلى - أسفل. ولاستعمال الأداء ، وعلى مطورى النظم والتطبيقات أن يعرفوا منهجية الشيء الموجه. لو أن مطور التطبيق استعمل الأداة العليا لهندسة البرمجيات المساعدة للحاسب أو أداة التحليل والتصميم، فإن الناتج سيكون رسمًا تخطيطيًا مطبوعًا أو توثيقيًا للنموذج الشيئى الموجه الذى تم إنشاؤه. وأما فى حالة استعمال مطور تطبيق الأداة المتكاملة لهندسة البرمجيات المساعدة للحاسب فإنه يبدأ بإنشاء نموذج الشيء الموجه، ومن ثم تولد الأداة التشفير الخاص بالتطبيق . ولكن أدوات الشيء الموجه المتكاملة لهندسة البرمجيات المساعدة للحاسب OO I- CASE قد تكون ذات نواتج بسيطة ، بحيث تسمح لمطور التطبيق بإنشاء رسوم تخطيطية ثابتة وتحريرها، ومن ثم ينشأ أقل تشفير ضرورى لتشغيل التطبيق . من ناحية أخرى ، تعرض الرسوم التخطيطية بيئة تطوير مفسرة تسمح لمطور التطبيق أن يدخل البرامج (الطرق) methods والقيم وتنفيذ الرسائل messages ، مع تطوير التطبيق بطريقة لولبية أو تكرارية. وتوفر الأداة المتكاملة لهندسية البرمجيات المساعدة للحاسب لغة برمجة وسيطة تسمح لمطور التطبيق أن يعدل الأداة نفسها إلى جانب توفير بيئة التحرير والإطارات واختيار الأداة [٢]. ومثالاً على الأداة المتكاملة لهندسة البرمجيات المساعدة للحاسب المتكاملة هى "تقنية نمذجة الشيء " Object Medeling Techinique (OMT) ومثالاً على الأداة العليا لهندسية البرمجيات المساعدة للحاسب هى "تحليل الشيء الموجه وتصميم الشيء الموجه" OO Analysis and OO Design. وبالمقارنة بين منهجيات تحليل وتصميم الشيء الموجه التى كانت بين هندسة برمجيات الشيء الموجه (OOSE) وتقنية نمذجة الشيء (OMT) وتحليل الشيء الموجه (OOA) وتصميم الشيء الموجه هرميًا (HOOD) والتصميم الهيكلى للشيء الموجه (OOSD) وتصميم استنتاج المسئولية (RDD) - وُجِدَ أن كلاً من تقنية الشيء الموجه OOSE، هندسة برمجيات الشيء الموجه OMT من

أفضل أنواع هندسة البرمجيات المساعدة للحاسب والتى يمكن الاعتماد عليها لعمليات تحليل وتصميم الشىء الموجه.

تقنية نمذجة الشىء OMT :

إنه يشبه معظم منهجيات تحليل وتصميم الشىء الموجه التى تحاول تبسيط عملية التحليل والتصميم وتحقيق القدرة على إعادة الاستخدام وانعكاس العالم الحقيقى. وهكذا ويمكن تلخيص معالم "تقنية نمذجة الشىء" OMT كالآتى[٢] :

١- أنه أسلوب منهجى مدروس ويعتبر أحد أقدم المنهجيات فى تحليل وتصميم الشىء الموجه.

٢- أنه أسلوب منهجى مختبر فى مختلف التطبيقات.

٣- يمتلك تقنية نمذجة قوية مبنية على ثلاثة نماذج، هى:

* نموذج الشىء Object Model .
* النموذج المتحرك Dynamic Model .
* النموذج الوظائفى Functional Model .

٤- أنه يتمتع ببساطة وسهولة التحويل من مرحلة التحليل لمرحلة تصميم النظام لتصميم الشىء وأخيراً لمرحلة التنفيذ (التطبيق).

٥- أنه لديه القدرة على استخلاص الأنواع Classes وعلاقات الربط فيما بينهما من بداية المشكلة.

خطوات الأسلوب المنهجى لتقنية نمذجة الشىء :

الشىء هو التشييد الأساسى الذى يجمع كلاً من هيكل البيانات والسلوك فى كينونة واحدة. وجوهر تطوير الشىء الموجه هو تعريف وتنظيم مفاهيم نطاق التطبيق أكثر من التمثيل النهائى لتلك المفاهيم فى لغة برمجة الشىء الموجه. وتتلخص خطوات الأسلوب المنهجى فيما يلى:

أولاً - تحليل نظام التطبيق :

هدف التحليل هو تطوير نموذج لما سيفعله النظام. ويتم التعبير عن النموذج فى حدود الأشياء وعلاقات الربط وتدفق التحكم المتحرك dynamic Control Flow والتحويلات الوظيفية. وتظل عملية الاستحواذ على المتطلبات والتشاور مع محددى المتطلبات مستمرة أثناء التحليل.

١- التوصيف المبدئى للمشكلة :

- ينبغى الحصول على التوصيف المبدئى للمشكلة فى البدء.

٢- بنـاء نموذج الشىء :

يتم بناء نموذج الشىء بوضع رسم تخطيطى له مدعمًا بقاموس للبيانات عن طريق:

- تعريف أنوا ع الشىء.
- تعريف قاموس للبيانات بحيث يحتوى على توصيفات الأنوا ع، الخصائص والترابطات.
- إضافة الترابطات بين الأنوا ع.
- إضافة الخصائص للأشياء.
- تنظيم وتبسيط أنوا ع الشىء باستعمال الوراثة.
- اختيار تداول المسارات باستعمال السيناريوهات وتكرار الخطوات السابقة.
- تجميع الأنوا ع فى وحدات Modules مبينة على زوج مغلق ومترابط وظيفياً.

نموذج الشىء = رسم التخطيطى لنموذج الشىء + قاموس البيانات

يصنف نموذج الشىء الهيكل الثابت للأشياء فى النظام وعلاقات الربط بينهما. ويحتوى ونموذج الشىء على الرسوم التخطيطية للشىء . والرسم التخطيطى هو رسم ذو رؤوس nodes تمثل أنوا ع الشىء ، ووصلات arcs تمثل علاقات الربط بين الأنوا ع.

٣- تطوير النموذج المتحرك :

- إعداد سيناريوهات تساعد على تتابع التفاعل النموذجي.
- تعريف الأحداث بين الأشياء وإعداد حدث تقفى الأثر event trace لكل سيناريو.
- إعداد الرسم التخطيطي لتدفق الحدث في كل النظام .
- تطوير الرسم التخطيطي لحالة كل نوع له سلوك متحرك.
- مراجعة سياق Consistency وإتمام الأحداث المشاركة بين الرسوم التخطيطية للحالة.

النموذج المتحرك = الرسوم التخطيطية للحالة + الرسم التخطيطي لتدفق الحدث الشامل

ويصف النموذج المتحرك أشكال النظام التي تتغير عبر الزمن، ويتم استعماله لتوصيف أشكال تحكم النظام. ويتكون من الرسم التخطيطي للحالة والذي يتم التعبير عنه برؤوس تمثل الحالات ووصلات تمثل الانتقالات بين الحالات التي تحدث بسبب الأحداث.

٤- تركيب النموذج الوظيفي :

- تعريف قيم المدخلات والمخرجات
- استعمال الرسوم التخطيطية لتدفق البيانات حسب الاحتياج لإظهار التبعيات الوظيفية.
- وصف ما تفعله كل وظيفة.
- تعريف القيود.
- وصف الشروط المثالية.

النموذج الوظيفي = الرسوم التخطيطية لتدفق البيانات + القيود

يصف النموذج الوظيفي تحويلات قيم البيانات في النظام ، ويحتوي على الرسوم التخطيطية لتدفق البيانات . والرسم التخطيطي لتدفق البيانات رسم ذو رؤوس هي العمليات، وصلات هي تدفق البيانات.

٥- تحقيق صحة وتكرار وتنقية النماذج الثلاثة :

- إضافة عمليات المفتاح التى يتم اكتشافها أثناء إعداد النموذج الوظيفى لنموذج الشيء. وعدم إظهار كل العمليات أثناء التحليل؛ لأنها تتراكم بدون ترتيب لنموذج الشيء ، والعمل على إظهار العمليات الأكثر أهمية.

- التحقق من صحة الأنواع والارتباطات والخصائص والتأكيد من اتساق العمليات وإتمامها عند مستوى التجريد الذى يتم اختياره. ومقارنة النماذج الثلاثة بالمشكلة المبدئية وتوضيح معرفة النطاق واختيار النماذج باستعمال السيناريوهات.

- تطوير أكثر للسيناريوهات التفصيلية بتباينها مع السيناريوهات الأساسية ، مع استعمال سيناريوهات "ماذا لو" لتحقيق صحة النماذج الثلاثة بشكل أكثر.

- دمج الخطوات السابقة حسب الاحتياج لاكتمال التحليل.

توثيق التحليل = المشكلة المبدئية + نموذج الشيء + النموذج المتحرك + النموذج الوظيفى

ثانياً - تصميم النظام :

أثناء تصميم النظام ، ينبغى اختيار هيكل المستوى الأعلى للنظام. والنموذج Paradigm الشيئى الموجه لا يقدم تبصيرات خاصة فى تصميم النظام . وفيما يلى خطوات تصميم النظام المتبعة:

١- تنظيم النظام إلى نظم فرعية.

٢- تعريف التزامنية المورثة فى المشكلة.

٣- تخصيص النظم الفرعية للمعالجات والمهام.

٤- اختيار السياسة الأساسية لتطبيق تخزين البيانات فى حدود هياكل البيانات، الملفات وقواعد البيانات.

٥- تعريف المصادر الشائعة وتحديد الآليات اللازمة لمراقبة تداولهم.

٦- اختيار طريقة معينة لتطبيق مراقبة البرمجيات:

- باستعمال مكان فى البرنامج لمسك الحالة.

- أو باستعمال مهام التزامنيية.

٧- أخذ اعتبارات حدود الشروط فى الحسبان.

٨- إنشاء تناوب الأولويات.

تطوير تصميم النظام = هيكل البيئات الاساسية للنظام + قرارات سياسة المستوى الأعلى

تصميم الشيء :

يجب إتقان نموذج التحليل أثناء تصميم الشيء وتوفير الأساسات التفصيلية للتنفيذ، إلى جانب اتخاذ القرارات الضرورية؛ لتحقيق نظام بدون قصور فى التفاصيل الخاصة للغة ما أو نظام قواعد بيانات معينة. ويبدأ تصميم الشيء من اتجاهية العالم الحقيقى لنموذج التحليل نحو اتجاهية الحاسب المطلوب لتطبيق خاص:

١- الحصول على عمليات نموذج الشيء من النماذج الأخرى:

- إيجاد عملية Operation لكل معالجة فى النموذج الوظيفى.

- تعريف عملية لكل حدث فى النموذج المتحرك الذى يعتمد على تطبيق التحكم.

٢- تصميم الخوارزميات لتنفيذ العمليات:

- اختيار الخوارزميات التى تقلل تكلفة تنفيذ العمليات.

- اختيار هياكل البيانات الملائمة للخوارزميات.

- تعريف أنواع داخلية جديدة وعمليات حسب الضرورة.

- تخصيص مسئولية للعمليات التى لم تكن مرتبطة بشكل واضح مع نوع معين.

٣- تداول المسارات المثلى للبيانات:

- إضافة ارتباطات زائدة عن الحد لتقليل التداول وتعظيم الملاء مة.

- إعادة ترتيب حساب التعبيرات لزيادة الكفاءة.
- حفظ القيم المستنتجة لتجنب إعادة حساب التعبيرات المعقدة .

٤- تنفيذ مراقبة البرمجيات باستخدام الطريقة التى تم اختيارها لتصميم النظام.

٥- ضبط أو تعديل هيكل النوع لزيادة الوراثة:

- إعادة ترتيب وتعديل الأنواع والعمليات لزيادة الوراثة.
- تجريد السلوك الشائع من مجموعات الأنواع.
- استعمال التفويض لمشاركة السلوك حيث تكون الوراثة دلالياً غير صحيحة.

٦- تصمم تنفيذ الارتباطات:

- تحليل تبحر الارتباطات.
- تنفيذ كل ارتباط حسب الشيء المميز له أو بإضافة خصائص قيم الشيء الواحد أو لكل الأنواع فى الارتباط.

٧- تحديد التمثيل الدقيق لخصائص الشيء .

٨- تحزيم الأنواع والارتباطات فى وحدات Modules.

توثيق التصميم = نموذج الشيء التفصيلى + النموذج المتحرك التفصيلى + النموذج الوظيفى التفصيلى

بالرغم من تضمن ذلك الترتيب أهمية إلا أن:

- مطورى التطبيقات ذوى الخبرة قادرون على تجميع العديد من الخطوات أو تنفيذ خطوات معينة فى توازٍ لإجراء المشروع.
- تكرار الخطوات ضرورى فى المستويات الأقل تبعية للتجريد ، التى وتضيف أكثر التفاصيل للنموذج.
- بعد اكتمال التحليل عند المستوى الأعلى للتجريد، يمكن أن تصمم النظم الفرعية فى المشروع الضخم بشكل مستقل ومتزامن عند المستويات الأقل للتجريد.

- التمييز بين التحليل والتصميم قد يبدو فى بعض الأوقات اعتباطيًا وفوضويًا. القواعد البسيطة التالية ينبغى أن ترشد القرارات التى تركز على المجال المناسب للتحليل والتصميم.
- نموذج التحليل يشمل المعلومات ذات المعنى من الرسم المنظورى للعالم الحقيقى ويمثل المنظور الخارجى للنظام.
- نموذج التحليل ينبغى أن يكون قابلاً للفهم لمستخدم النظام ويوفر أساسًا مفيدًا لانتزاع الاتساق داخليًا وإمكانية تحقيقه.

وعلى العكس، فإن نموذج التصميم يتم استنتاجه بصلة وثيقة بتطبيق الحاسب. وهكذا نموذج التصميم يجب أن يكون ذا تأثير معتدل ويمكن تشفيره. وعمليًا فإن أجزاء كثيرة لنموذج التحليل وهذه الأجزاء يمكن أن تكون جاهزة للتطبيق بدون تغيير، وهكذا قد يوجد تداخل بين نموذجى التحليل والتصميم. نموذج التصميم يجب أن يعنون مستوى التفاصيل الذى يتم تجاهله فى نموذج التحليل. نماذج التحليل والتصميم تعمل على توفير توثيق ذى قيمة للنظام من نموذجين مختلفين ولكن متكاملين من جهة الرسوم المنظورية.

تشييدات نمذجة الشيء Object Modeling Constructs :

أهميـة النموذج الشيئى :

النموذج الشيئى هو أحد النماذج الثلاثة الأكثر أهمية. وهو بناء مؤكد للنظام حول الأشياء أكثر منه حول الوظائف؛ وذلك لأن الشيء الموجه أكثر قربًا لمناظرة العالم الحقيقى؛ ومن ثم يعتبر النموذج الشيئ أكثر مرونة فيما يتعلق بالتغيير. ويوفر النموذج الشيئى تمثيل رسم بديهى للنظام ويكون ذا قيمة للاتصال مع العملاء وتوثيق هيكل النظام. من ناحية أخرى، إن نماذج البيانات الشيئية الموجهة واضحة التكامل مع البرامج الشيئية الموجهة وأيضًا للتحويل من قواعد البيانات الشيئية الموجهة إلى قواعد البيانات العلاقية، حيث إنه وثيق الصلة بالتحويل كما سنرى.

المفاهيم الأساسية لنمذجة الشيء :

- الأشياء والأنواع :

الشيء Object هو أى شيء موجود وله هوية. حيث إن مجموعة من الأشياء المتشابهة تشكل نوع الشيء Object Class. والشيء هو واقعة لنوع يتم توصيفه بواسطة الخصائص أو الحقول.

نوع الشيء يتم تمثيله بواسطة صندوق box له ثلاثة أجزاء. وأجزاؤه الرئيسية تحتوى من أعلى إلى أسفل على : اسم نوع الشيء، قائمة الخصائص، قائمة العمليات . كل اسم خاصية قد يليه تفاصيل اختيارية مثل النوع والقيمة الافتراضية. هكذا اسم كل عملية قد يتبع بتفاصيل اختيارية مثل قائمة الأدلة ونوع الناتج. وقد تحذف العمليات من تخطيطات الرسم[٣].

الصناديق فى الشكل رقم (١٠- ١) تشير إلى أنواع الأشياء. على سبيل المثال: نوع الشيء آلات Equipment له الخصائص التالية:

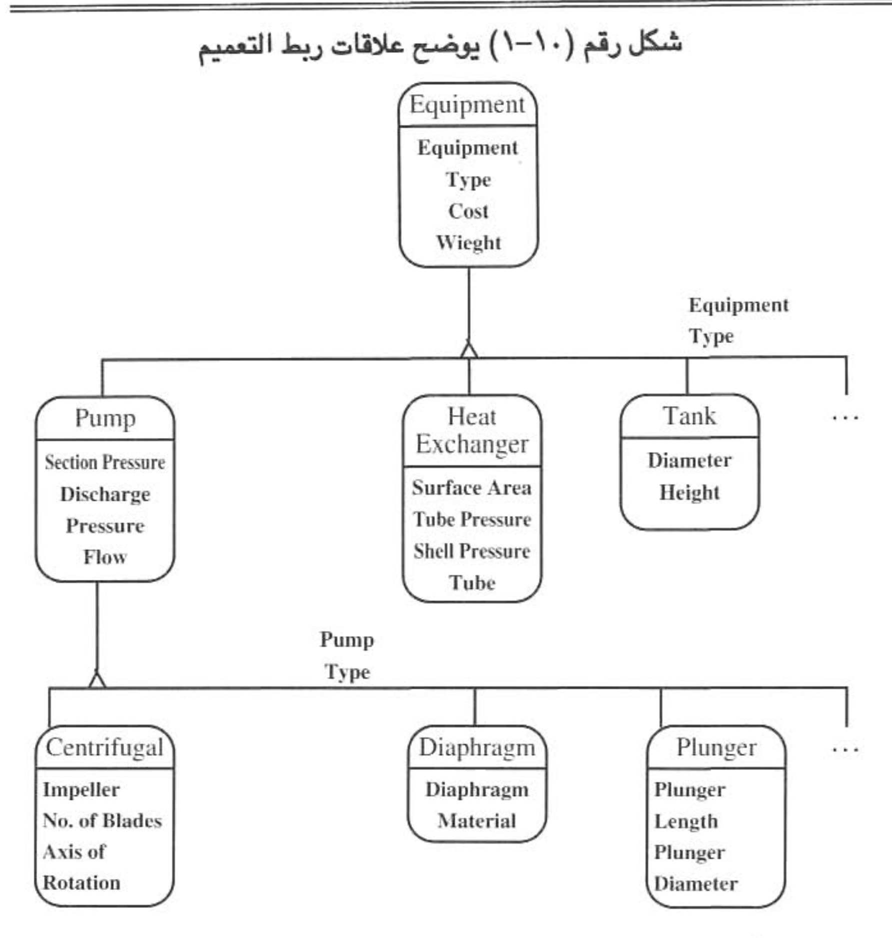
اسم الآلة . Equipment Name

التكاليف . Cost

الوزن . Weight

وهكذا كما هو موضح بالشكل خصائص كل نوع شيئى.

شكل رقم (١٠-١) يوضح علاقات ربط التعميم



- علاقات الربط Relationships :

علاقة الربط هى رابطة منطقية بين الأشياء. وهناك ثلاثة أنوا ع من علاقات الربط بين الأشياء[٤].

* التعميم .
* التجميع .
* الارتباط .

حيث تستخدم رموز خاصة في نهايات خط علاقة الربط تشير إلى عدد الأشياء لنوع واحد يرتبط بكل نوع شيء آخر. وهذا يسمى التعددية لعلاقة الربط multiplicity of the relationship وهي مشابهة للتعددية التي سبق الإشارة إليها في النموذج العلاقي Cardinality. على سبيل المثال: دائرة صغيرة صماء تعني العديد. والعديد في هذا السياق هو صفر أو أكثر. أما دائرة صغيرة مجوفة hollow تعني صفرًا أو واحدًا. نهاية الخط المستقيم للعلاقة بدون رمز يشير إلى واحد بالضبط.

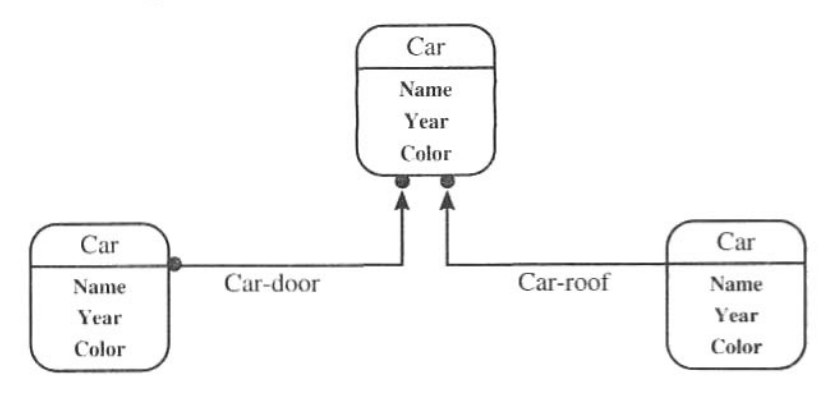
أ- علاقة ربط التعميم Generalization Relationship :

أجزاء علاقة ربط التعميم أو علاقة الربط "يكون" is-a نوع شيء لأنواع فرعية مقصورة بشكل تعاوني. التعميم قد يكون لدية عدد اعتباطي من المستويات . في الشكل رقم (١٠-١) المثلثات ترمز للتعميم. ويأتي في قمة التعميم، الآلات EQUIP-MENT وهي نوع أصلي Superclass ، أما الطلمبة Pump والمبدل الحراري heat ex-changer وحوض البترول tank فهي أنوا ع فرعية. وهكذا تكون الخصائص مورثة من المستوى الأعلى إلى المستوى الأدنى.

ب- علاقة ربط التجميع Aggregation Relationship :

علاقة ربط التجميع أو علاقة الربط "جزء من " a-part-of هي محتوى تجميعي، ويجمع التجميع أشياء المستوى الأقل في أشياء مركبة Composite objects. التجميع قد يكون متعدد المستويات. وكما في الشكل رقم (١٠-٢) فالرف roof هو "جزء من" a-part-of السيارة Car، العديد من الأبواب doors هي "جزء من" a-part-of السيارة Car. في هذه الحالة السيارة هي تجميع ، أما الرف والأبواب فهي محتويات التجميع وغالبًا يعرض تبعية موجودة تشبه التعميم.

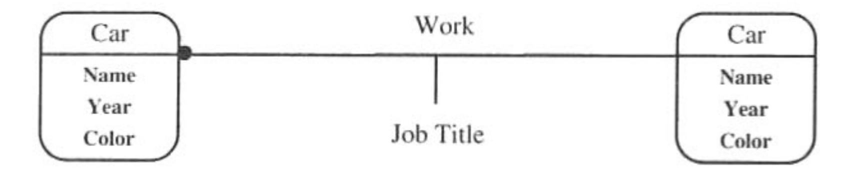
شكل رقم (١٠ – ٢) يوضح علاقات ربط التجميع



ج – علاقة ربط الارتباط Association Relationship :

تربط علاقة ربط الارتباط بين نوعين أو أكثر للأشياء المستقلة. والارتباط لا يعرض تبعية موجودة existence dependency كما فى الشكل رقم (١٠–٣)، فإن العديد من الموظفين Employees يعملون فى الشركة Company والموظف Employee يدير موظفين آخرين employees.

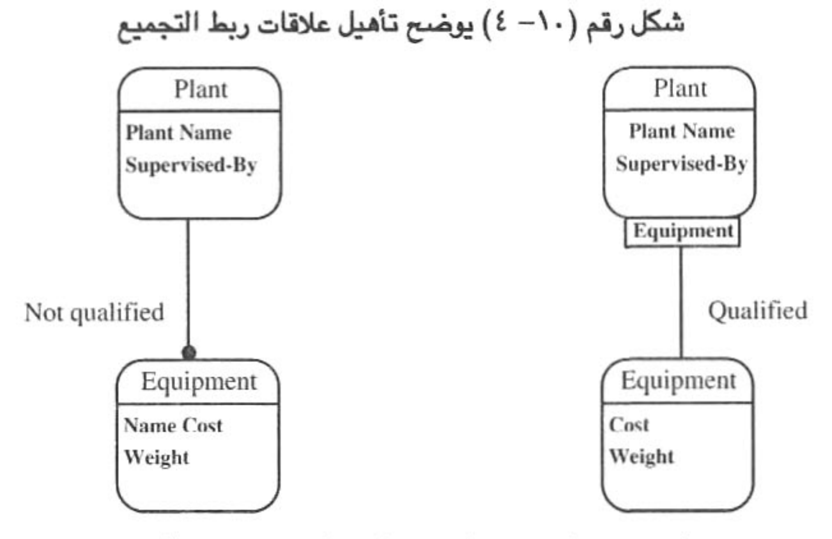
شكل رقم (١٠–٣) يوضح علاقات ربط الارتباط



د– تأهيل العلاقات Qualification of Relationship :

يضيف التأهيل معلومات عن جانب علاقة الربط التى تعبر عن عديد. فى الشكل رقم (١٠–٤) يوجد تجميع ، تأهيل وبدون تأهيل . على سبيل المثال: فالمصنع Plant له العديد من قطع الآلات Pieces of equipment التى يتم تمييزها بواسطة الاسم؛ ومن ثم يعتبر اسم الآلة Equipment name هو حقل التأهيل.

شكل رقم (١٠- ٤) يوضح تأهيل علاقات ربط التجميع



التحويل من النموذج الشيئى الموجه إلى النموذج العلاقى :

يعد المثال Paradigm الشيئى الموجه تقنيات متعددة البراعات Versatile. فهو لا يوفر فقط الأساسى الصوتى لتصميم النظم ولكن يمكن أيضاً أن يستعمل لتصميم قواعد البيانات. والتصميمات الشيئية الموجهة تتمتع بالكفاءة والتماسك وأقل ميلاً لحدوث مشاكل التى تنزل كوارث بتقنيات تصميم قواعد البيانات الأخرى. كفائدة جانبية يفضل استعمال تقنية التصميم الموحد لتحسين تكامل قاعدة البيانات ، ولكن بسبب بساطة الأساس الرياضى لنموذج البيانات العلاقى، فإن قواعد البيانات العلاقية قد أثبتت نجاحاً خارقاً فى دعم تطبيقات معالجة البيانات النمطية ، مثل المحاسبة ، والمرتبات وغيرها؛ ولذا فإن قواعد البيانات العلاقية قد زاد نطاق انتشارها واتسعت تقنية استعمالها فى بداية عقد الثمانينيات؛ مما يدفع إلى التفكير فى كيفية التحويل من النموذج الشيئى الموجه إلى النموذج العلاقى.

قواعد التحويل من النموذج الشيئى الموجه إلى النموذج العلاقى:

النموذج الشيئى هو أحد مكونات تقنية نمذجة الشىء OMT الذى سبق توضيحه. وهو ما سيتم استخدامه فى التحويل من الهياكل الشيئية إلى الجداول العلاقية. والجداول الناتجة تتجه نحو الشكل الطبيعى الثالث 3NF.

والشكل الطبيعي الثالث هو الفائدة الجوهرية لنمذجة الشيء. أحدها قد يقابل الشكل الطبيعي الأول 1NF عند تفكيك الأشياء المعقدة. وامتداد هذا التفكيك يعتمد على المقابلة للذرية (القيم الفردية) والتطبيق.

وإنه من السهل رؤية الجداول العلاقية المستنتجة من الشيء أن تحقق الشكل الطبيعي الثاني 2NF . ولكن معظم انتهاكات الشكل الطبيعي الثالث 3NF المرئية تحدث عندما تدخل معلومات غير جوهرية (غريبة) في جدول أو وجود جدول مفتقر إلى التركيز(٥).

(١) تحويل الأنواع إلى الجداول العلاقية Converting classes to Tables :

- كل نوع يتحول مباشرة إلى جدول علاقي. كل حقول الشيء تصبح خصائص للجداول العلاقية.
- كل شيء له هوية لا تتكرر ID. كل المرجعيات للأشياء تتم عن طريق هويه الشيء. وتكون هوية الشيء في الرسم التخطيطي للشيء ضمنية ويجب أن توضع في الجداول العلاقية بشكل صريح.
- هذا المستوى للتحويل يحكم استعمال القيم الخالية . وتعني القيم الخالية أن قيمة الخاصية غير معروفة أو غير قابلة للتطبيق للقيم المرتبة الملحقة بها. الخصائص في المفاتيح المرشحة يجب ألا تكون خالية.

شكل رقم (١٠-٥) تحويل النوع Class إلى جدول علاقي

Plant
- Plant Name
- Plant Address
- Supervised By

object Model

Plant

Attribute	*Null?*	*Domain*
Plant ID	*N*	*ID*
Plant Name	*N*	*Long Name*
Plant Address	*Y*	*Address*
Supervised By	*Y*	*Pers. Name*

Candidate Keys: (Plant ID) (Plant Nane)
Frequently Accessed : (Plant ID)

- كل خاصية أيضًا لها نطاق أو فئة لقيم شرعية للخاصية. وإنه من غير المرغوب فيه أن يكون للخاصية نطاق نوع معين فى جدول علاقى ونوع مختلف فى جدول علاقى آخر. ومفهوم النطاق يشابه النوع الأساسى القوى فى لغة البرمجة. فى شكل رقم (١٠-٥) قائمة بالمفاتيح المرشحة وأيضًا قائمة بمجموعات الخصائص التى يحتمل بالخبرة تداولها بشكل متكرر. هذه المجموعات أولية Prime ويتم استهدافها للفهرسة أو التفريم. والهويات والأسماء يجب أن تكون متواقعة لكى تكون مرجعيات شائعة.

- ويجب أن يأخذ فى الحسبان أن الأشياء فى نوع معين قد تجزأ أفقيًا أو رأسيًا أو الاثنان معًا. على سبيل المثال: لو أن نوعًا معينًا له وقائع عديدة قليلاً منها يشار إليه غالبًا، فى هذه الحالة التقسيم الأفقى قد يحسن الكفاءة بوضع الأشياء المتداولة بشكل متكرر فى جدول علاقى واحد. والأشياء المتبقية فى جدول علاقى آخر كما هو مبين فى شكل رقم (١٠-٦أ). وبالطبع فإن التطبيق سوف لا يستفيد من التقسيم الأفقى ما لم يعرف أى الجداول العلاقية يجب أن يتم بحثه، لو أن النوع له خصائص بأشكال تداول مختلفة قد تساعد على تقسيم الشىء رأسيًا كما هو مبين فى الشكل رقم (١٠-٦ب).

شكل رقم (١٠-٦) التقسيم الأفقى و الرأسى للجداول العلاقية

شكل رقم (١٠-٦أ) التقسيم الأفقى للجداول العلاقية

Plant ID	Plant Name	Supervised By	Plant Address
1	PA	AI- Sammak	AlGubail
3	PB	Al-Mobark	AlDhahran

Plant ID	Plant Name	Supervised By	Plant Address
80	PA	AI- Sulaiman	Al khobar

شكل رقم (١٠-٦ب) التقسيم الرأسى للجداول العلاقية

Plant ID	Plant Name	Supervised By
1	PA	AI- Sulaiman
3	PB	Al-Mobark
80	PA	AI- Sulaiman

Plant ID	Plant Name	Plant Address
1	PA	Al Gubail
3	PB	Al Dhahran
80	PA	AI Khobar

(٢) تحويل التعميم والتخصيص إلى جداول علاقية :

هناك أربعة طرق لتحويل علاقات الربط تعميم وتخصيص إلى جداول علاقية :

- **الطريقة الأولى** : "الطريقة المعيارية" حيث إن النوع الأصلى والنوع الفرعى كليهما يحول إلى جدول علاقى. وهوية الشىء عبر التعميم يتم حمايتها خلال استعمال الهوية المشتركة. ويمثل هذا أفضل أسلوب لتناول علاقات الربط تعميم وتخصيص الذى يعرض وراثة متعددة من فصل الأنواع عن بعضها البعض ، جدول علاقى للنوع الأصلى وجدول (جداول) علاقى للأنواع الفرعية. ويوضح الشكل رقم (١٠-٧) الآلية العامة حيث إنه يشاهد فى الشكل الطبيعى الثالث 3NF .

- **الطريقة الثانية** : "طريقة النوع الفرعى المتعدد" وتتجاهل هذه الطريقة الجدول العلاقى للنوع الأصلى وتكرر كل خصائص النوع الأصلى فى كل جدول علاقى مناظر للنوع الفرعى. وفى هذه الطريقة يشاهد الشكل الطبيعى الثالث 3NF ولكن أقل تحقيقاً من "الطريقة المعيارية" السابقة.

– **الطريقة الثالثة البديلة** : طريقة "جدول علاقى واحد للنوع الأصلى". وهذه الطريقة تجلب كل خصائص النوع الفرعى إلى مستوى النوع الأصلى. وهذه الطريقة تنتهك الشكل الطبيعى الثالث 3NF ؛ على سبيل المثال هوية خاصية الآلية Equipment ID وخاصية اسم الآلة Equipment name يكون حقل المفتاح الأساسى ولكن قيم الخاصية أيضًا يعتمد على نوع الآلة Equipment Type.

– **الطريقة الأخيرة** : لتحويل التعميمات إلى جداول علاقية لتناول الوراثة المتعددة لتداخل الأنواع هى أن يتم استعمال جدول علاقى واحد النوع الأصلى، وجدول علاقى آخر للأنواع الفرعية وجدول علاقى ثالث علاقة ربط التعميم.

شكل رقم (١٠–٧) تحويل التعميم والتخصيص إلى جداول علاقية

Equipment
Equipment
Name
Cost
Weight

Equipment

Tank
Diameter
Height

. . .

Object Model

EQUPME

Attribute Name	*Null?*	*Domain*
Equipment ID	*N*	ID
Equipment Type	*N*	Long
Equipment	*N*	Name
Name	*Y*	Equip
Cost	*Y*	Type
Weight		Dollars

Candidate Keys: (Equip ID) Equip Name)
Frequently Accessed : (Equip ID)

TANK

Attribute Name	*Null?*	*Domain*
Equipment ID	*N*	ID
Diameter	*Y*	Length
Height	*Y*	Length

Candidate Keys: (Equip ID)
Frequently Accessed : (Equip ID)

(٣) تحويل التجميعات إلى جداول علاقية :

علاقات الربط متعدد - لمتعدد من الضرورة أن تحول إلى جداول علاقية مميزة ، وهذا نتيجة للشكل الطبيعى NF. وعلاقات الربط واحد - لواحد ، وواحد - لمتعدد قد تحول إلى جداول علاقية مميزة أو تدمج مع الشىء المشارك. وتناول تجميعات واحد- لواحد ، أو واحد - لمتعدد تعتمد على السياق.

التجميعات التابعة الوجود Existence-dependent يتم دمجها مع الجدول العلاقى (الشيئى) لتبسيط إلزامية السلامة . فى السياق للشكل رقم (١٠-٨) ، كل جزء للآلة Piece of equipment يجب أن يتم تخصيصه لمبانى وتجهيزات المصنع Plant. التجميعات الحرة Free-Standing يتم تخزينها في جداول علاقية مميزة. فى الشكل رقم (١٠-٩) كل نوع رف type of roof تستعمل لعديد من نماذج السيارة Car models ، الرف هو الجزء الذى يوجد مستقلاً لأى سيارة.

شكل رقم (١٠- ٨) دمج التجميعات التابعة الوجود Existence-dependent مع الجدول العلاقى

Plant
- **Plant Name**
- **Plant Address**
- **Supervised By**

Equipment

Equipment
- **Cost**
- **Weight**

...

Object Model

PLANT

Attribute Name	*Null?*	*Domain*
Plant ID	*N*	ID
Plant Name	*N*	Long
Plant Address	*Y*	Name
Supervised By	*Y*	Address

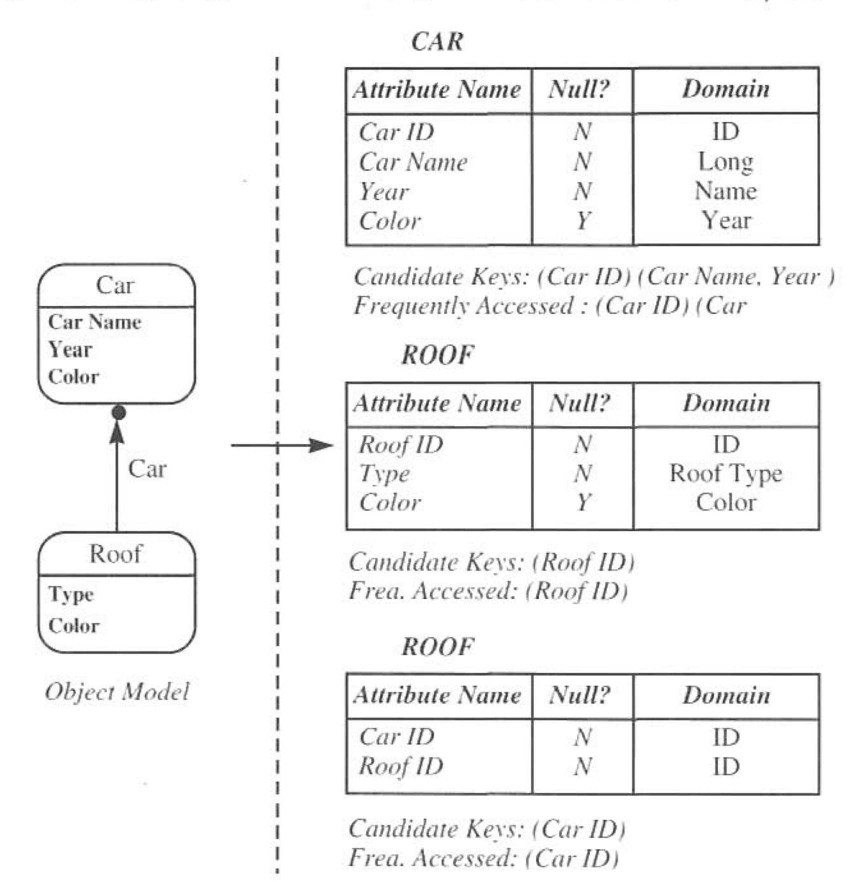
Candidate Keys: (Plant ID) (Plant Name)
Frequently Accessed : (Plant ID)

EQUPME

Attribute Name	*Null?*	*Domain*
Equipment ID	*N*	ID
Plant ID	*N*	ID
Equipment	*N*	Long
Name	*Y*	Name
Cost	*Y*	Dollars
Weight		Weight

Candidate Keys: (Equip ID) (Plant ID)
(Equip ID)

شكل رقم (١٠-٩) تخزين التجميعات الحرة Free-Standing فى جداول علاقية مميزة



CAR

Attribute Name	*Null?*	*Domain*
Car ID	*N*	ID
Car Name	*N*	Long
Year	*N*	Name
Color	*Y*	Year

Candidate Keys: (Car ID) (Car Name, Year)
Frequently Accessed : (Car ID) (Car

ROOF

Attribute Name	*Null?*	*Domain*
Roof ID	*N*	ID
Type	*N*	Roof Type
Color	*Y*	Color

Candidate Keys: (Roof ID)
Frea. Accessed: (Roof ID)

ROOF

Attribute Name	*Null?*	*Domain*
Car ID	*N*	ID
Roof ID	*N*	ID

Candidate Keys: (Car ID)
Frea. Accessed: (Car ID)

(٤) تحويل الارتباطات إلى جداول علاقية :

طبقًا للقواعد، تحول الارتباطات إلى جداول علاقية مميزة كما هو فى الشكل رقم (١٠-٨). صفات الارتباط تصبح خصائص للجدول العلاقى المناظر للارتباط. الارتباطات لا تطوى مع الشيء كما فى الشكل رقم (١٠-٦)، ماعدا أداء عنق الزجاجة. ويوجد عديد من الأسباب للارتباطات الخارجية كما يلى:

١- ارتباطات بين أشياء مستقلة لوزن تركيبى متساوٍ. وبصفة عامة ، تبدو ملائمة لتلويث الأشياء بمعلومات لأشياء أخرى.

٢- طوى الارتباطات بصفات توصيفية فى الأشياء؛ مما يؤدى إلى انتهاك الشكل الطبيعى الثالث 3NF .

٣- أنه من الصعب الحصول على التعددية الصحيحة حيث إن ارتباطات واحد-لواحد ، واحد - لمتعدد قد تكون خارجية عندما تكون علاقات ربط الارتباطات إجبارية، وارتباطات متعدد - لمتعدد يجب ان تكون خارجية.

٤- التمثيل التماثلى يبسط البحث والتعديل.

بصفة عامة عند إنشاء جدول علاقى مميز للارتباط تصبح المفاتيح الأساسية للأنواع المرتبطة وأى خصائص وصل خصائص للجدول العلاقى للارتباط. والجدول العلاقى للارتباط يضع دائمًا المفاتيح الخارجية من الشىء المرتبط إلى الشىء غير الخالى.

ملخص قواعد التحويل المنطقى من النموذج الشيئى إلى النموذج العلاقى للبيانات :

النماذج الشيئية هى أكثر واقعية لكونها قابلة للتطبيق فى قواعد البيانات الشيئية الموجهة . وبعض الخطوات المضمنة لتحويل نماذج البيانات الشيئية الموجهة إلى نماذج بيانات علاقية، هى كالآتى:

(١) تحويل الأنواع إلى جداول علاقية :

- كل نوع يحول إلى جدول علاقى أو أكثر.
- هوية الشىء تصبح المفتاح الأساسى للجدول العلاقى.
- كل الخصائص الأخرى تصبح خصائص مناظرة للجدول العلاقى .

(٢) تحويل الارتباطات إلى جداول علاقية :

- كل ارتباط متعدد - لمتعدد يحول إلى جدول علاقى متميز (واضح المعالم) distinct.

– كل ارتباط واحد – لواحد أو واحد – لمتعدد يحول إلى جدول علاقى مميز عندما تكون علاقة ربط الارتباط اختيارية ، وبخلاف ذلك يذوب كمفتاح خارجى فى الجدول العلاقى للنوع المتعدد.

– الارتباطات ذات التعددية N-ary تحول إلى جداول علاقية متميزة .

– الارتباط المؤهل يحول إلى جدول علاقى متميز بثلاث خصائص على الأقل ، أحدها الخاصية المؤهلة.

(٣) تحويل التجميعات إلى جداول علاقية :

(أ) تحول تعميم الوراثة الفردية إلى جدول علاقى :

– النوع الأصلى وكل نوع فرعى يحول إلى جدول علاقى.

– عدم تكرار الجدول العلاقى الخاص بالنوع الأصلى وخصائص لكل جدول علاقى خاص لنوع فرعى.

– عدم جلب الجدول العلاقى الخاص بالنوع الأصلى خصائص جدول علاقى خاص بنوع فرعى إلى جدول علاقى للنوع الأصلى فى المستوى الأعلى له.

(ب) تحويل الوراثة المتعددة المنفصلة والمتداخلة إلى جداول علاقية .

– النوع الأصلى وكل نوع فرعى يحول إلى جدول علاقى .

– إضافة إلى الوراثة المتعددة المتداخلة ، فإن علاقة ربط التعميم أيضًا تحول إلى جدول علاقى.

مثال (١٠–١):

يبين نموذج كينونة – علاقة المطور EER لتطبيق مستشفى hospital application فى شكل رقم (١٠–١٠) قائمة بأنوا ع الكينونات والخصائص وعلاقات الربط[٦]. سوف يتم تمثيله بعد ذلك باستخدام النموذج الشيئى Object Model والذى يعد أحد مكونات تقنية نمذجة الشىء OMT كما هو مبين فى الشكل رقم (١٠–١١).

أولاً – قائمة بأنواع الكينونات والخصائص :

١- نوع كينونة الطبيب الجراح Surgeon لها الخصائص التالية : الاسم Name والعنوان Address ورقم التليفون No–Tel.

٢- نوع كينونة الطبيب الاستشارى Consultant وهى تمثل نوعًا فرعيًا لنوع كينونة الطبيب الجراح Surgeon . كل طبيب استشارى Consultant يكون متخصصًا فى فرعًا معين للجراحة ، ويتم تسجيل ذلك كخاصية إضافية تسمى التخصص Specialty.

٣- نوع كينونة المريض Patient لها الخصائص التالية:

رقم المريض #Patient (لكل مريض رقم لا يتكرر) والاسم Name والعنوان Address ، رقم التليفون Tel-No ، تاريخ الميلاد BirthDate ، النوع Sex وفصيلة الدم blood-Group.

٤- نوع كينونة المريض الخاص Private-Patien وهى نوع فرعى لنوع كينونة المريض Patient. وتمثل خاصية رقم الغرفة #Room ورقم غرفة المريض التى ستخصص له.

٥- نوع كينونة المرضة Nurseلها الخصائص التالية : رقم الهيئة #Staff ،(رقم الهيئة لا يتكرر) ،الاسم Name ، العنوان Address ، رقم التليفون Tel-No ، النوع Sex ، ومرتبة المرضة Grade. المرضة قد تختص إما بجناح Ward أو بغرفة عمليات Theatre.

٦- نوع كينونة الجناح Ward لها الخصائص التالية: رقم الجناح #Ward ، (رقم الجناح لا يتكرر) ، نوع الجناح Ward-Type ، وعدد الأسرة No-of-Beds.

٧- نوع كينونة غرفة العمليات Theatre لها الخصائص التالية : رقم غرفة العمليات # Theatre (رقم غرفة العمليات لا يتكرر)، ونوع غرفة العمليات Theatre-Type.

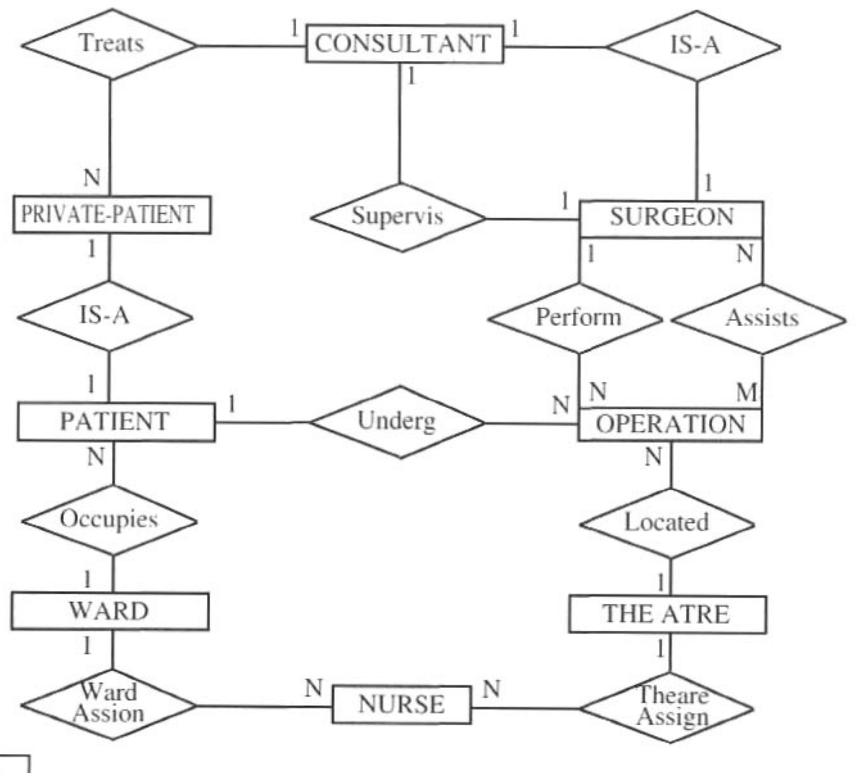
٨- نوع كينونة العملية Operation لها الخصائص التالية : نوع العملية Operation-Type ، تاريخ إجراء العملية Date والوقت إجراء العملية Time.

ثانيًا – قائمة بعلاقات الربط الملائمة:

١- علاقة الربط "ينفذ" Performs ، وهى علاقة ربط واحد – لمتعدد بين نوع كينونة الطبيب الجراح Surgeon مع نوع كينونة العملية Operation لكونها عضواً إجبارياً لهذه العلاقة. وهذا يعنى أن كل عملية يجب أن يتم تنفيذها بواسطة طبيب جراح.

٢- علاقة الربط "يساعد" Assists وهى علاقة ربط متعدد – لمتعدد بين نوع كينونة الطبيب الجراح Surgeon ونوع كينونة العملية Operation والتى تتضمن هؤلاء الأطباء الجراحين الذين يساعدون فى كل عملية . ويمكن إلحاق خاصية دور Role بعلاقة الربط تلك ، والتى تمثل الدور الذى يلعبه كل طبيب جراح.

شكل رقم (١٠-١٠) نموذج كينونة – علاقة ER لقاعدة بيانات مستشفى (افتراضية)



٣- علاقة الربط "يشرف" Supervises هى علاقة ربط واحد - لمتعدد بين نوع كينونة الطبيب الاستشارى Consultant ونوع كينونة الطبيب الجراح Surgeon. وتكون مجموعة أعضاء نوع كينونة الطبيب الجراح فى هذه العلاقة اختيارية حيث قد يوجد طبيب جراح (مثل الطبيب الاستشارى) غير مكلف بالإشراف.

٤- علاقة الربط "يعالج" Treats هى علاقة ربط واحد - لمتعدد بين نوع كينونة الطبيب الاستشارى Consultant ونوع الكينونة الفرعى للمريض الخاص Private-Patient. ومجموعة أعضاء نوع كينونة المريض الخاص فى هذه العلاقة تكون إجبارية. وهذا يعنى أن كل مريض خاص يخصص له طبيب استشارى لمعالجته.

٥- علاقة الربط "يجرى عملية" Undergoes هى علاقة ربط واحد - بين نوع كينونة المريض Patient ونوع كينونة عملية Operation. نوع كينونة عملية هى عضو إجبارى لهذه العلاقة حيث إن العملية دائمًا قد ترتبط بالمريض.

٦- علاقة الربط "يشغل" Occupies هى علاقة ربط واحد - لمتعدد بين نوع كينونة الجناح Ward ونوع كينونة المريض Patient حيث إن مجموعة أعضاء نوع كينونة مريض غالبًا ما تكون إجبارية حيث إن معظم.

٧- المرضى يخصص لهم جناح عند دخول المستشفى.

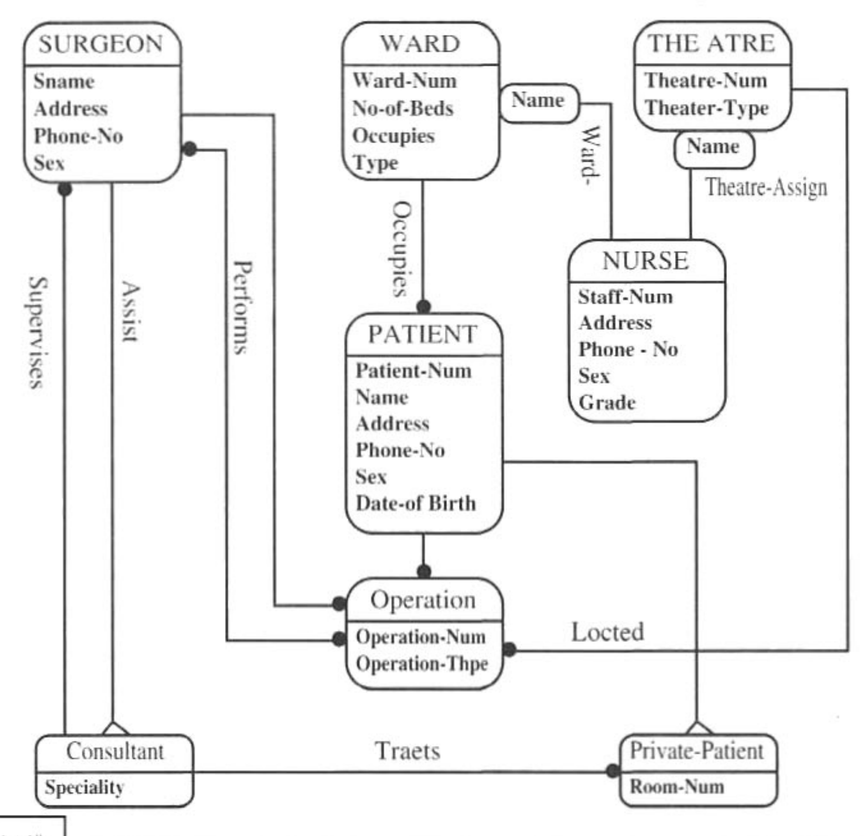
٨- علاقة الربط "مقطنة" Located هى علاقة ربط واحد-لمتعدد بين نوع كينونة غرفة العمليات Theatre ونوع كينونة العملية Operation لنوع كينونة عملية هى عضو إجبارى لهذه العلاقة حيث إن كل عملية يجب أن تكون مقطنة بشكل واضح فى غرفة عمليات.

٩- علاقة الربط "يخصص جناح" Ward-Assign هى علاقة ربط واحد - لمتعدد بين نوع كينونة جناح Ward ونوع كينونة ممرضة Nurse. ويمكن إلحاق خاصية تاريخ التخصيص Data-Assigned التى تمثل تاريخ تخصيص ممرضة معينة لجناح معين لهذه العلاقة. ونوع كينونة ممرضة هى عضو اختيارى لهذه العلاقة حيث إن الممرضة قد تكون مخصصة لجناح فى الوقت المعطى وقد لا تكون.

١٠- علاقة الربط "تخصص غرفة عمليات" Theatre-Assign هى علاقة ربط واحد-لمتعدد وهى علاقة ربط اختيارية بين نوع كينونة غرفة العمليات Theatre ونوع كينونة الممرضة Nurse ، ويلحق بها خاصية تخصيص التاريخ Data-Assined التى تعطى تاريخ تخصيص ممرضة معينة لغرفة العمليات.

١١- علاقة الربط "يكون" IS-A وتوجد هذه العلاقة بين نوع الكينونة الفرعى الطبيب الاستشارى Consultant ونوع كينونة الطبيب الجراح Surgeon . وتوجد علاقة ربط أخرى بين النوع الفرعى "مريض خاص" Private-Patient ونوع كينونة "المريض" Patient.

شكل رقم (١٠-١١) النموذج الشيئى لقاعدة بيانات مستشفى (افتراضية)



إحدى فوائد نماذج قواعد البيانات الشيئية الموجهة هى تكامل التصدير المباشر مع البرامج الشيئية الموجهة. وبصفة عامة، فإنه من الصعب أن يصرح بتفاعل قاعدة بيانات متجانسة مع تشفير الإجراء . يساعد الاستعمال الشائع لنموذج الشىء المجاز Metaphor وتدوين نفس التصميم لنماذج البيانات والبرامج فى هذه الحالة Situation.

والهدف الرئيسى لاستعمال تطبيق المستشفى هو تحويله من نموذج قاعدة البيانات الشيئية الموجهة إلى نموذج قاعدة البيانات العلاقية لذلك أنه أكثر تعبيراً أن يتم كتابته فى صورة خوارزميات أو تشفير كاذب Pseudo Code. وهكذا فيما يلى الخوارزميات المكتوبة بمثل لغة البسكال Pascal-Like:

```
Class Surgeon
Properties
   Name, address, phone-no : String ;
   Sex:(Male , female);
   Supervised-by: Consultant;
      Inverse (exist) is consultant;
   Performs : Set (Operations)
      Inverse is Operation. Performed-by;
   Assists-at: Set (Operation)
      Inverse is Operation  , assisted-by;
Operations
   Create (--);
   Assign-duties(--);
End Surgeon.
```

```
Class Consultant

Inherit Surgeon

Properties
   Supervises : Set (Surgeon)
      Inverse is Surgeon supervised-by
   Treats : Set (Private-Patient)
      Inverse is Private-Patient.treated-by
Operations
   Create (--);
   Calculate-fees(hours,..);
End Consultant.


Class Patient
Properties
   Number: integer;
   Name , address , phone-no : string;
   Sex: ( male , female);
   Date-of-birth : date;
   Blood-group: (A,B,AB,O);
   On-ward : ward
      Inverse is ward.Patients;
   Undergoes : Set (operation)
      Inverse is Operation-Performed-on;
Operations
   Create (--)
   Admit (--)
   Discharge (--)
End Patient.
```

```
Class Private-Patient
Inherit Patient
Properties
   Room # : integer;
   Insurance : insurance-type;
   Treated-by : Consultant
      Inverse is consultant-treats;
Operations
   Create (--);
   Calculate-charge (--);
End Private-patient.

Class Ward
Properties
   Ward # : integer;
   No-of-beds : integer;
   Occupancy : integer;
   Type : (Geriatric , Pediatric ; Maternity,...);
   Nurses : Set (nurse)
      Inverse is nurse. Ward-assign;
Operations
   Create (--);
End Ward
```

```
Class Operation
Properties
   date : Date:
   type : Operation type ;
   Performed-on : Patient
      Inverse is Private-Operations;
   Performed-by : surgeon
      Inverse is surgeon. Performs;
   Assisted-by: Set (Surgeon)
      Inverse is surgeon. Assists-at;
   Located-in: Theatre
      Inverse is theatre. Holds;

Operations
   Create (--):
   Schedule (--)
   Cancel (--)
End Operation.

Class Nurse
Properties
   Staff# , name , address , phone# : String;
   Sex : (Male , Female);
   Grade :(Student, SEN,SRN,...),
   Ward-assign : ward
      Inverse is Ward-nurses;
   Theatre-assign: Theatre
      Inverse is Theatre-nurses;
Operations
   Create (--)
End Nurse
```

```
Class Theatre
Properties
  Theatre# : integer;
  Type : theatre type;
  Nurses : Set (Nurse)
    Inverse is Nurse-theatre-assign;
  Holds: Set (operation)
    Inverse is Operation-located-in;
Operations
  Create (--);
End theatre
```

طبقًا لقواعد التحويل الملخصة فى الجزء السابق ، فإن تحويل تطبيق المستشفى من نموذج قاعدة البيانات الشيئية الموجهة إلى نموذج قواعد البيانات العلاقية سيتم كالآتى:

(١) التحويل من الأنواع إلى مخططات الجداول العلاقية :

* SURGEON (Surgeon-ID, Sname, C-Name, Address,Phone-No, Sex).

* PATIENT (Patient-ID, Patient-Num, Ward-Num, Pname, Address , Phone-No, BirthDate , Sex , Blood-Type).

* NURSE (Nurse-ID, Staff-Num , Nname , Address , Phone-No , BirthDate, Sex , Blood-Type).

* THEATRE (Theatre-ID, Theatre-Num, Theatre-Type).

* OPERATION (Operation-ID, Operation-Num, Sname, Theatre-Num, Patient-Num, Operation-Type, Date) .

(٢) التحويل من التعميمات إلى مخططات الجداول العلاقية :

* CONSULTANT (Sname , Specialty) .

* PRIVATE-PATIENT (Patient-Num , Sname , Room-Num).

(٣) التحويل من الارتباطات إلى مخططات الجداول العلاقية :

* الارتباطات متعدد - لمتعدد يجب أن تحول إلى جداول علاقية متميزة :

Assists (Operation-Num, Sname, Role).

* الارتباطات واحد - لمتعدد قد تحول أو لا تحول إلى جداول علاقية طبقًا لعلاقة الربط بالارتباطات لحالة جدول علاقى معين . وهذا يعنى أن الارتباط قد يكون عضواً إجبارياً أو اختيارياً لجدول علاقى معين أو لا يكون.

(أ) الارتباط الإجبارى لجدول علاقي ليس له جدول علاقي متميز ولكن يتم تمثيله بإذابة مفتاحه الخارجى داخل ذلك الجدول العلاقى.

* علاقة الربط "يشغل" Occupies يتم تمثيلها باستعمال المفتاح الخارجى "رقم الجناح" ward-Num فى الجدول العلاقى "مريض" Patient حيث إن علاقة الربط هى غالبًا إجبارية للجدول العلاقى "مريض" Patient.

* علاقة الربط "يعالج" Treats التى يتم تمثيلها باستعمال المفتاح الخارجى "اسم الطبيب الاستشارى" Sname فى الجدول العلاقى "مريض خاص" Private-Patient.

*** علاقات الربط التالية:**

"ينفذ" Perfarms ، "يجرى عملية" Undergoes ، "مقطنة" Located يتم تمثيل تلك العلاقات باستعمال المفاتيح الخارجية فى الجدول العلاقى "عملية" .Operation والمفاتيح الخارجية التى يتم إذابتها فى ذلك الجدول العلاقى هى:

"اسم الطبيب الاستشارى" Sname ، "رقم المريض" Patient-Nun ، "رقم غرفة العمليات" Theatre-Num على التوالى.

(ب) الارتباط الاختيارى للجدول العلاقى قد يكون جدولاً علاقياً متميزاً:

* علاقات الربط "يخصص جناح" Ward-Assign "وتخصص غرفة عمليات" Theatre-Assign يتم تمثيل كل منها بواسطة بمخطط الجدول العلاقى المنفصل الذى يحتوى على خصائص المفتاح لفئات الكينونة التى تشارك معًا بالإضافة إلى خاصية

"تخصيص تاريخ" Date-Assign . كذلك علاقة الربط الاختيارية "يشرف" Supervises يتم تمثيلها باستعمال مخطط الجدول العلاقى المنفصل كالاتى:

* WARD-ASSIGN (Staff-Num , Ward-Num , Date-Assign)

* THEATRE-ASSIGN (Staff-Num , Theatre-Num , Date-Assign)

* SUPERVISES (Sname, C-Sname).

مقارنة بين نماذج البيانات الشيئية الموجهة والعلاقية :

الاستحقاقات النسبية للطرق العلاقية والشيئية الموجهة لإدارة البيانات يمكن تلخيصها تحت المواضيع التالية[٦] :

* أنواع البيانات.

* سلامة البيانات.

* تطوير المخطط.

* معالجة البيانات.

(١) أنواع البيانات Data Types :

يوجد فى نظم إدارة قواعد البيانات العلاقية نوع واحد فقط بصفة عامة لهيكلة البيانات ، يسمى نوع الجدول العلاقى. لا تكون خصائص الجداول العلاقية بعد التطبيع قابلة للتفكيك وهذه الخصائص تتجه إلى نوع بيانات بسيطة نسبياً وهى أنواع البيانات الأساسية (الأرقام الصحيحة - الأرقام الحقيقية - السلاسل الحرفية - وهكذا) . أما عن العمليات على الجداول العلاقية فتكون محصورة على الاسترجاع وتحديث الجداول المعرفة بواسطة قيم الخصائص.

فى قواعد البيانات الشيئية الموجهة يتم تخزين تعريفات النوع والمتغيرات لهذه الأنواع . وتعريفات النوع هى مناظرات المخططات فى نظم قواعد البيانات العلاقية. وبالمعالم الإضافية المهمة فإن الأنواع تكبسل سلوك الشىء بتغليف العمليات مع هيكل

البيانات. والنوع هو نفسه واقعة لنوع آخر (نوع وسط Metaclass) وقد يعالج بوصفه شيئًا مثل أى شىء آخر وأيضًا قد يمرر كمعلم Parameter لعملية ما.

(٢) سلامة البيانات Data Integrity :

تتطلب النظم العلاقية اتباع قواعد سلامة مرجعية لكى تكون ملزمة كما سبق وورد فى الفصل الرابع. ومع ذلك ، فإن النموذج العلاقى يكون عاجزًا عن التعبير عن قيود السلامة بالمحتويات الدلالية الأكبر من السلامة المرجعية للتصدير المباشر.

وعلى النقيض ، فى النموذج الشيئى الموجه ، فالنوع يعرف تجريد البيانات التى تشمل توصيف للعمليات (البرامج Methods) التى يمكن أن تطبق على وقائع النوع . مثل هذا التجريد يتم إنجازه فى درجة عالية من الاستقلالية. وهذا يعنى أنه من الممكن أن يتغير الأسلوب الذى يتم فيه تطبيق النوع بدون تأثير على الأنواع الأخرى أو برامج المعاملات التى تصنع استعمال التجريد . وكل الأشياء (متغيرات النوع) لها هوية لا تتكرر كما سبق وتم إيضاحه فى الفصل السادس . وهذا على النقيض من النموذج العلاقى حيث إن صفات الكينونة يجب أن تكون كافية لتمييزها عن الكينونات الأخرى.

(٣) تطور المخطط Schema Evalution :

نظم إدارة قواعد البيانات العلاقية تعرض مجالاً محدودًا لتطور المخطط نتيجة لعرض تسهيلات محدودة جدًا لتمديد أو تعديل هياكل البيانات الموجودة. وتتضمن تغيرات أنواع النطاق عادة إعادة كتابة الجداول العلاقية التى تتضمنها.

النموذج الشيئى الموجه يعرض مجالاً أكبر إلى حد بعيد لتطور المخطط خلال التمديد وتنقية هياكل البيانات الموجودة وتأثير إعادة الاستخدام لتشفير التطبيقات.

(٤) معالجة البيانات Data Manipulation :

بالرغم من الفوارق المعنوية فى نماذج البيانات ، فإن قواعد البيانات الشيئية قد يتم تفسيرها بطريقة مشابهة لنظم قواعد البيانات العلاقية . على سبيل المثال فعمليات الاختيار Selection العلاقية التى بواسطتها تحقق القيم المرتبة tuples شرطًا معينًا ، يتم اختيارها من جدول علاقى يكون مكافئًا لاسترجاع وقائع نوع خاص. ومع ذلك فإن

معظم الاستفسارات المعقدة على سبيل المثال التى تتضمن الربط أو وضع عمليات على أنواع متداخلة لا يتم تعرفها جيداً فى قواعد البيانات العلاقية. ولازال حتى الآن لا يوجد نموذج استفسار معرف جيداً لقواعد البيانات الشيئية الموجهة التى تعرض أقوى لغات قواعد البيانات.

تحليل وتصميم أداة تصميم قواعد البيانات الشيئية الموجهة :

يتم عمل تحليل وتصميم لأداة تصميم قواعد البيانات الشيئية الموجهة باستخدام تقنية نمذجة الشىء OMT. وأداة تصميم قواعد البيانات الشيئية الموجهة هى نظام برمجي يمكن تصميمه بشكل خاص؛ لكى يجعل نشاط تصميم قواعد البيانات الشيئية الموجهة أتوماتيكيًا، وهو الأمر الذى يمكن الاستفادة منه فى تحويل قواعد البيانات التقليدية أو غيرها إلى قواعد بيانات شيئية موجهة. وكانت هناك محاولات عديدة من قبل منتجى البرمجيات أن ينتجوا أداة لتصميم قواعد بيانات شيئية موجهة لكن المعضلة الرئيسية فى تلك العملية كانت عدم وجود منهجية معيارية. لذلك فإن محاولة تطوير أداة لعملية تصميم قواعد بيانات شيئية موجهة يمكن أن تقام على منهجية يتم بناؤها على نماذج البيانات الشيئية الموجهة والأدوات المختلفة للنظم التقليدية والشيئية الموجهة .

عملية تصميم قواعد البيانات الشيئية الموجهة :

يدير مصمم قاعدة البيانات فى أثناء عملية التصميم كل إجراءات عملية التصميم، حيث يتم إنشاء النموذج المبنى على المتطلبات. وتبزغ عملية التصميم كالآتى:

عمليات النموذج :

وفيها يقوم مصمم قاعدة البيانات بإنشاء نموذج جديد أو فتح نموذج سابق . وفى حالة إنشاء النموذج الجديد يجب إعطاؤه اسماً.

(١) إنشـاء الأنواع :

ينبغى على مصمم قاعدة البيانات تعريف الأنواع التى سيتم استخدامها أثناء عملية التصميم.

(٢) توصيف الخصائص :

يقوم مصمم قاعدة البيانات بتوصيف الخصائص الخاصة بكل نوع.

(٣) توصيف البرامج (الطرق) :

يقوم مصمم قاعدة البيانات بتوصيف البرامج (الطرق) الخاصة بكل نوع.

(٤) إعداد الوصلات :

يقوم مصمم قاعدة البيانات بتعريف الوصلات بين الأنوا ع وأنوا ع هذه الوصلات.

(٥) الوقائع :

على مصمم قاعدة البيانات اختيار التصميم الذى تم إعداده بإلحاق بعض الوقائع لكل نوع تم إنشاؤه؛ مما يجعل المصمم يعيد النظر فى الأنوا ع من منظور واقعى لا من منظور مفاهيمى فقط.

(٧) اهتمامات متنوعة :

على مصمم قاعدة البيانات تخزين النموذج بين الحين والآخر لحفظ نموذج العمل. ووضع إمكانية عمل نسخة احتياطية وكذلك إعادة التسمية والحذف للنموذج الذى تم إنشاؤه . كما أنه لا بد من توفير إمكانية طباعة كل تفاصيل النموذج على الطابعة أو ملف بالإضافة إلى عرض صفات النموذج عند الحاجة إليها.

خطوات دورة حياة عملية تصميم الأداة :

الهدف من أداة تصميم قاعدة البيانات الشيئية الموجهة هو توفير نظام عمل أتوماتيكى بشكل تام تقريبًا قادر على إنشاء ومراقبة وإدارة عملية تصميم قاعدة البيانات الشيئية الموجهة بكفاءة. ويتعامل مستخدم أداة التصميم مع واجهة تطبيق مدعمة بالرسم تسمح بالمدخلات النصية خلالها التى يجب أن تصمم على نحو ودى على قدر الإمكان. وعملية التصميم تتكون من خطوات دورة الحياة التالية:

(١) ينشئ مصمم قاعدة البيانات نموذجًا جديداً مع كتابة اسم وتوصيف ذلك النموذج.

(٢) يعرف مصمم قاعدة البيانات الأنواع الخاصة بالنموذج والخصائص والبرامج (الطرق) الملحقة بها.

(٣) تعريف مكونات كل خاصية :

- اسم الخاصية (على النظام التأكد من عدم تكراره).
- النوع الأساسى للخاصية.
- توصيف الخاصية إذا استدعت الحاجة إلى ذلك.

(٤) تعريف مكونات البرنامج (الطريقة) :

* اسم البرنامج (على النظام التأكد من عدم تكراره).
* تعريف المعلمات الداخلة والخارجة للبرنامج (إن وجدت).
* توصيف المعلمات إذا استدعت الحاجة إلى ذلك.
* توصيف الخوارزميات الخاصة بكل برنامج (طريقة).

(٥) تعريف الوقائع لكل نوع كنموذج أولى Prototype للنوع المفاهيمى.

(٦) تعريف الوصلات للأنواع التى تم إنشاؤها. كل وصلة تصل نوعين. وهناك ثلاثة أنواع للوصلات التى يمكن أن تستعمل هى:

Gen-Spec , Whale-Part , Message Connection

(٧) ينبغى أن يوفر النظام للمصمم مكتبة عامة للأنواع التى تساعده على استخلاص نوع أو أكثر منها لاستعمالها فى النموذج الجارى إنشاؤه. هذه الأنواع التى يتم استخلاصها يجب أن تكون قابلة للمعالجة (أى بإضافة وحذف خصائص وبرامج منها وإليها).

(٨) ينبغى أن يوفر النظام تسهيلات لطباعة التفاصيل للنموذج الذى تم إنشاؤه.

(٩) يجب أن يوفر النظام أسلوباً لحذف الأنواع والوصلات بأسلوب مناسب . وينبغى على النظام أن يعيد رسم النموذج بعد حذف أى نوع أو وصلة . وعلى النظام أن يأخذ فى الحسبان اعتبارات أنواع الوصلات المختلفة المدعمة له ، على سبيل المثال نوع وصلة مثل " Gen-Spec " عند حذف نوع Gen فإن كل أنواع Spec ينبغى أن يتم حذفها .

تحليل أداة تصميم قاعدة البيانات الشيئية الموجهة :

ينبغى تحليل أداة تصميم قاعدة البيانات الشيئية الموجهة بقصد الاستفادة باستعمالها فى عملية تصميم قاعدة البيانات الشيئية الموجهة .

(أ) النموذج الشيئى :

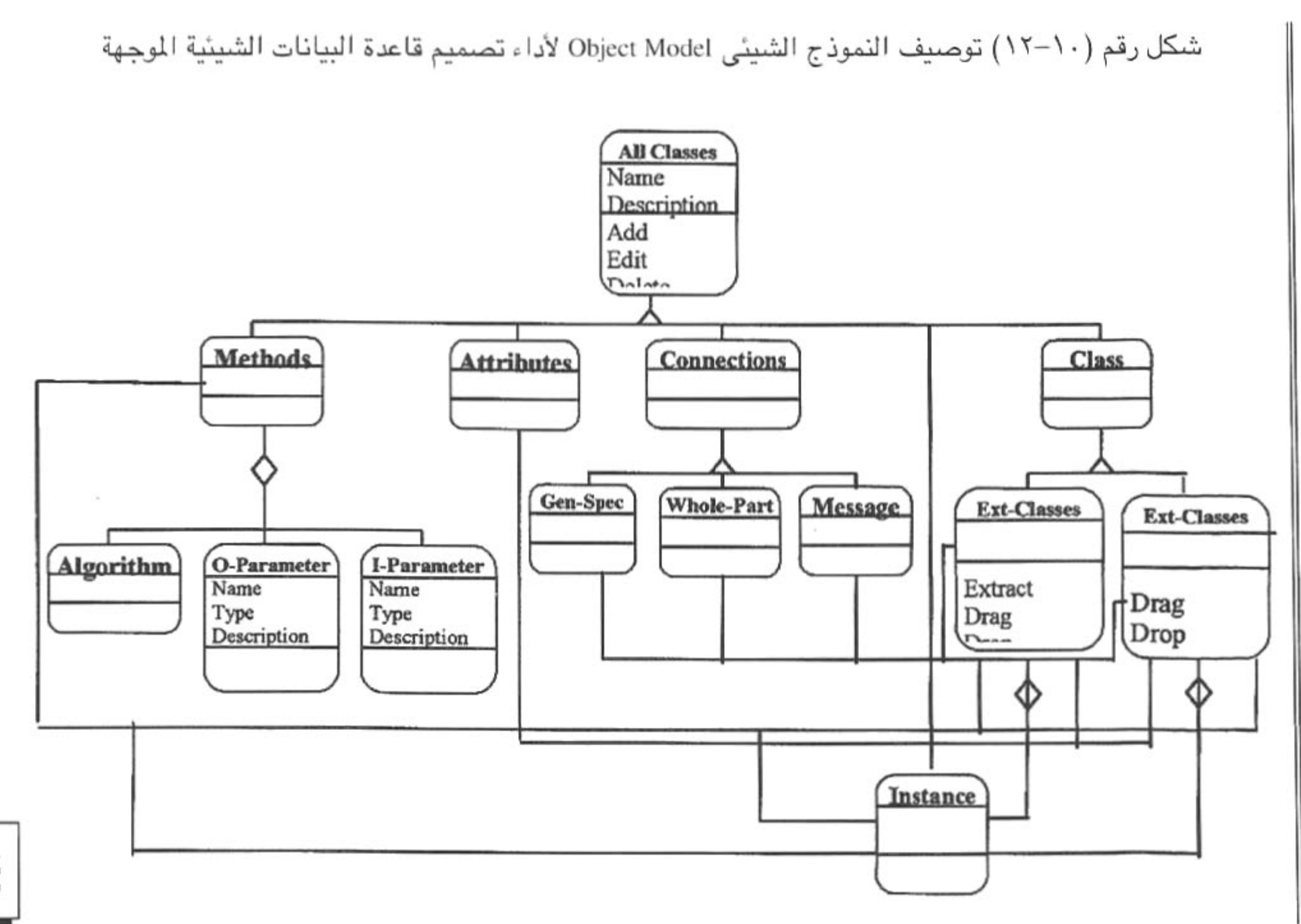
تتبع خطوات التحليل المذكورة فى بداية هذا الفصل والتى تساعد على استخلاص الأشياء من المتطلبات ومن معرفة بعض النماذج الخاصة بقواعد البيانات الشيئية الموجهة ونظم تصميم الشىء الموجهة ، وبملاحظة المصطلحات الواردة فى خطوات دورة حياة عملية تصميم الأداة التى يتم استعمالها فى النموذج الشيئى. إن عديداً من الصفات التى تصف الأشياء يتم الحصول عليها بالشعور البديهى لما ينبغى أن يشبه الشىء وكيفية سلوكه المتوقع.

*** تعريف أنواع الشىء :**

فيما يلى قائمة بأنواع الشىء المرشحة:

- الأنواع.
- الخصائص.
- البرامج (الطرق).
- الوقائع.
- الوصلات.
- مكتبة عامة للأنواع.

شكل رقم (١٠-١٢) توصيف النموذج الشيئى Object Model لأداء تصميم قاعدة البيانات الشيئية الموجهة



وسوف يتم تتبع هذه الأنواع خلال التحليل التالى:

الأنواع :

هى المشاركات فى أداة تصميم قاعدة البيانات الشيئية الموجهة. ويمكن للأنواع أن تمثل بأشياء متنوعة للنموذج؛ لذا لا بد من تحديد أى من الصفات تكون مهمة لكل أنواع الشىء بغض النظر عن التطبيق.

الخصائص :

هى الصفات التى قد يمتلكها كل نوع. وكل خاصية لها قيمة لكل واقعة شىء.

البرامج (الطرق) :

هى الخدمات أو العمليات التى قد يمتلكها كل نوع - ويمكن أن تطبق البرامج على النوع نفسه أو على نوع آخر خلال الوصلة.

الوقائع :

هى نماذج أولية للأنواع المفاهيمية. وكل نوع يعرف بواسطة المستخدم أو يستخلص من المكتبة العامة له الوقائع التى يمكن للمستخدم أن يحاكى بها الأشياء الحقيقية.

الوصلات :

هى أسلوب لتوصيل الأنواع بعضها ببعض . وقد أخذ فى الحسبان ثلاثة أنواع للوصلات قد سبق ذكرها. وكل نوع له آلية خاصة فى توصيل الأنواع.

مكتبة النوع العامة :

تحتوى على أنواع مجردة ، يمكن أن تستخلص أثناء عملية التصميم بواسطة المصمم. هذه الأنواع يمكن أن تخصص بواسطة المصمم فى أثناء عملية التصميم.

*** تعريف الارتباطات والخصائص :**

وفيما يلى تعريف الارتباطات والخصائص فى النموذج:

النــوع :

كل نوع له اسم ، وتوصيف، وموقع . والأنواع أيضًا لها خصائص وبرامج (طرق) سوف يتم تجزئتها فى أشياء منفصلة.

الخصائـص :

تحتاج الخصائص إلى معرفة الأنواع التى ترتبط هى معها وما الوقائع الخاصة بها يصف المستخدم اسم ونوع وتوصيف كل خاصية.

البرامج (الطرق) :

تحتاج البرامج إلى معرفة الأنواع التى ترتبط معها، وما هى وقائعها. ويصف المستخدم اسم وتوصيف كل برنامج. وكل برنامج له معلمات تعمل كمدخلات ومخرجات يجب تعريف أسمائها وأنواعها. مع مراعاة توصيف الخوارزم المتعلق بكل برنامج (طريقة).

مكتبة النوع العامة :

تشبه النوع ولكن أنواعها التى يتم تعريفها مسبقًا والتى سوف تلحق بها يجب تمييزها عن الأنواع المنشأة بواسطة المستخدم.

الوقائع :

تحتاج إلى معرفة الأنواع التى ترتبط هى معها . كل واقعة سوف يكون لها قيم لكل خاصية ، وأسماء الطرق (البرامج) التى يتم تطبيقها عليها.

*** تعريف العمليات :**

الأنواع :

تحتاج الأنواع بالتأكيد إلى عمليات إنشاء وتحرير وحذف وسحب وإلقاء. فى حالة حذف النوع الأصلى (الانواع المرتبطة خلال نوع وصلة Gen-Spec) فان كل الأنواع الفرعية ينبغى أن تحذف فى نفس الوقت.

الخصائص :

هناك العديد من العمليات الخاصة بالخصائص مثل إضافة وتحرير وحذف خاصية.

البرامج (الطرق) :

وتعد أيضاً العمليات الخاصة بالبرامج مثل إضافة وتحرير وحذف برنامج .

الوصلات :

تكمن العمليات المتعلقة بالوصلات في توصيل وصلة بين نوعين وتحرير الوصلة بين نوعين وحذف الوصلة بين نوعين.

الوقائع :

كذلك العمليات الخاصة بالوقائع مثل إضافة وتحرير وحذف واقعة.

المكتبة النوع العامة :

تكمن العمليات الخاصة بمكتبة النوع العامة فى استخلاص النوع من المكتبة. ومع ذلك لو أن النوع تم استخلاصه ، فإن عمليات التحرير والحذف يتم تطبيقها على النوع المستخلص.

*** تعريف الوراثة :**

الأنواع :

كل الأنواع السابق ذكرها (الأنواع - الخصائص - البرامج - الوصلات - الوقائع - مكتبة النوع العامة) لها عمليات إضافة وتحرير وحذف ، والتى يتم تجميعها تحت كل أنواع النوع الأصلى التجريدى.

الخصائص :

النوع ومكتبة النوع العامة لها نفس الخصائص بالإضافة إلى عمليات التحرير والحذف. ويتم تجميع النوع ومكتبة النوع العامة تحت النوع الأصلى التجريدى.

الوصلات :

الوصلات الثلاثة (Gen-Spec , Whole-Part , Message) يتم تجميعها تحت وصلة النوع التجريدى.

البرامج (الطرق) :

البرامج لها الخوارزميات والمعلمات الخاصة بها (المدخلات والمخرجات) والتى يتم تجميعها بواسطة نوع البرامج.

الارتباطات :

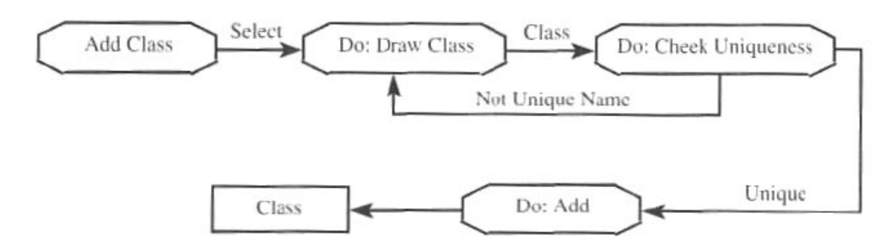
الارتباط بين نوعين (اللذين يتم تعريفهما واستخلاصهما بواسطة المستخدم) وأنواع الوصلات المتنوعة والخصائص والبرامج ينبغى أن تؤخذ بعين الاعتبار.

(ب) النموذج المتحرك :

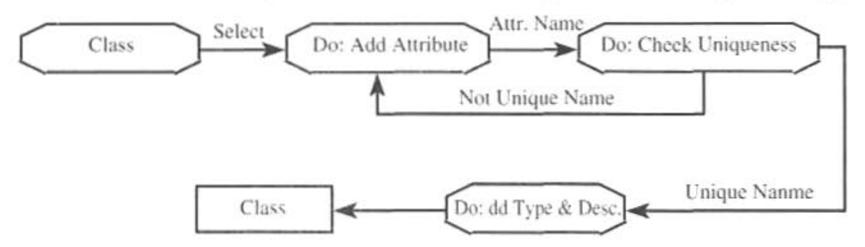
حيث إن أداة تصميم قاعدة البيانات الشيئية الموجهة هى أداة تفاعلية؛ لذا فإن النموذج المتحرك وواجهة تطبيق المستخدم من الضرورة أن يكونا لهما نفس الشىء. وفيما يلى النموذج المتحرك لأداة تصميم قاعدة البيانات الشيئية الموجهة والذى يمكن توضيحه كما فى الشكل رقم (١٠-١٣) باستعمال تدوينات تقنية نمذجة الشىء OMT.

شكل رقم (١٠-١٣) توصيف النموذج المتحرك Dynamic Model

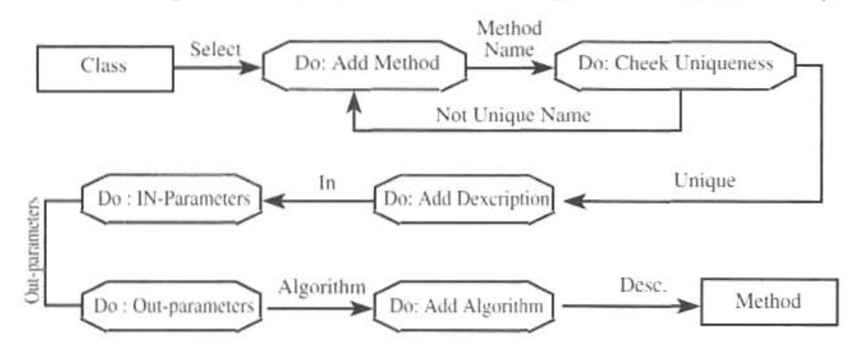
شكل رقم (١٠-١٣أ) توصيف النموذج المتحرك Dynamic Model لإضافة نوع Class



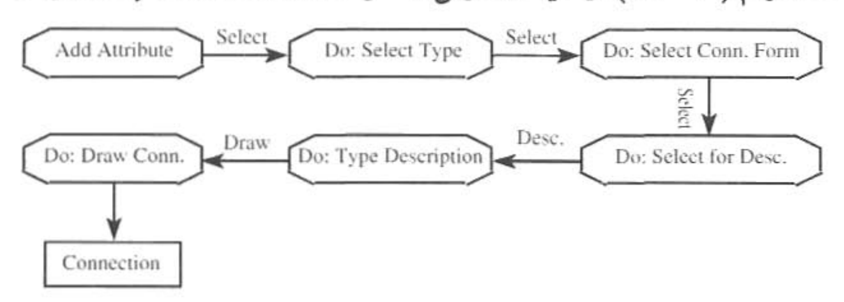
شكل رقم (١٠-١٣ب) توصيف النموذج المتحرك Dynamic Model لإضافة خاصية Attribute



شكل رقم (١٠-١٣جـ) توصيف النموذج المتحرك Dynamic Model لإضافة برنامج (طريقة) Attribute



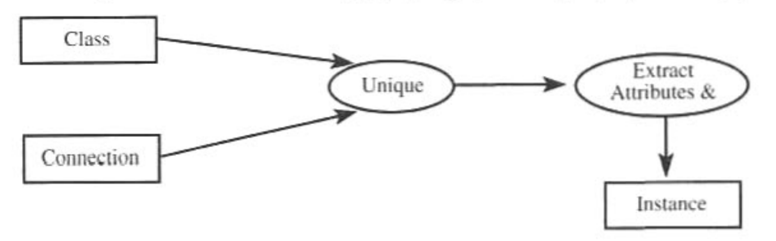
شكل رقم (١٠-١٣د) توصيف النموذج المتحرك Dynamic Model لإضافة وصلة



(ج) النموذج الوظيفى :

النموذج الوظيفى لمعظم الحسابات المكثفة يكون للوراثة والوقائع. يوصِّف الشكل رقم (١٠-١٤) النموذج الوظيفى للوقائع.

شكل رقم (١٠-١٤) توصيف النموذج الوظيفى Functional Model للواقعة Instance



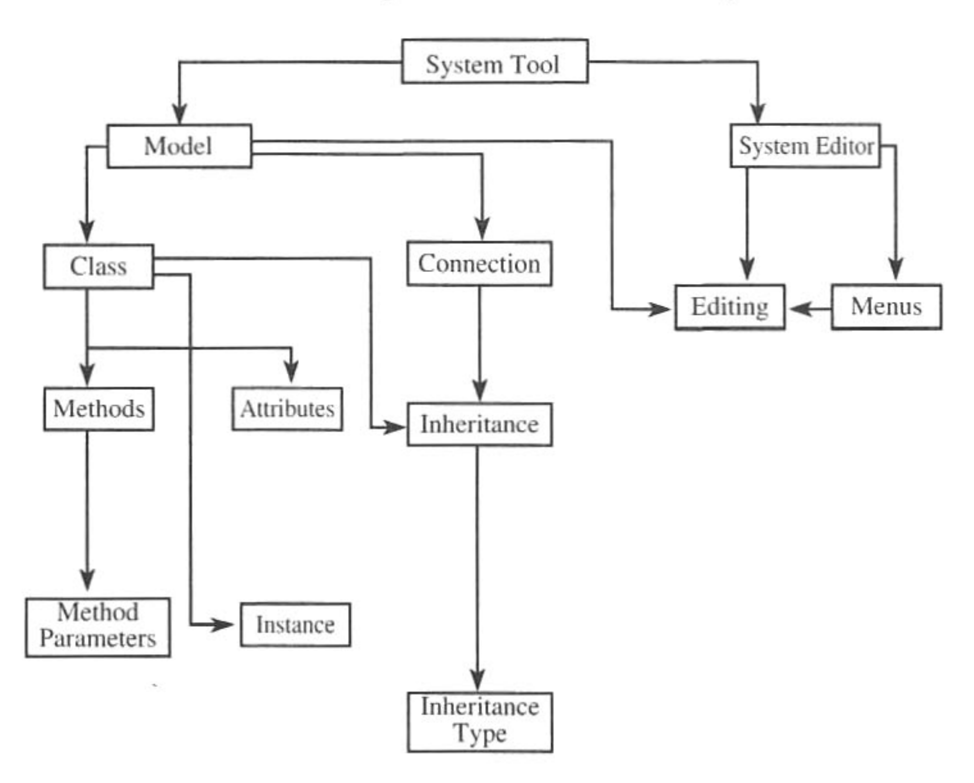
تصميم النظام :

تركيب النظام System Architecture بجميع الأشياء فى النظام طبقًا لخطوات عملية تصميم قاعدة البيانات الشيئية الموجهة التى تم توصيفها مسبقًا. ويبين الشكل (١٠-١٥) تركيب أداة تصميم قاعدة البيانات الشيئية الموجهة.

تفاعل أداة تصميم قاعدة البيانات الشيئية الموجهة يسمح للمستخدم بتصميم قاعدة بيانات شيئية موجهة باستعمال واجهة تطبيق مدعمة برسومات توضيحية graphical interface . بتفاعل أداة تصميم قاعدة البيانات الشيئية الموجهة، ويمكن للمستخدم إنشاء الأنواع، واستعمال مختلف أنواع الوصلات والوقائع.

يحتوى النموذج على أنواع ووصلات مختلفة. وكل نوع له خصائص وبرامج متنوعة. كل برنامج يحتوى على معلمات خاصة به. ويجب أن ترتبط الوراثة بنوع الوصلات ونوع الوراثة . ومحرر النظام System Editor يحتوى على القوائم وسلوك المؤشر mouse الذى يتحكم فى كل عملية تحرير فى النظام. وعملية التحرير يجب أن تتضمن الإضافة والحذف والنسخ والقص واللصق ... وغيرها.

رقم (١٠-١٥) البناء المعماري لنظام أداة تصميم قاعدة بيانات شيئية موجهة
Architecture of Object-Oriented Database Tool System



الهوامش :

1- [Hurson 1993] A. R. Hurson, Simin H. Pakzad and Jia-Bing Cheng. 'Object-Oriented Database Management Systems : Evolution and Perfomance Issues', **IEEE Computer**, February 1993.

2- [Bertino 1992] Elisa Bertino, Mauuro Negri, Giuseppe Pelagatti and Lisia Sbattella. 'Object-Oriented Query Languages : The Notions and the Issues', IEEE **Transaction on Knowledge and Data Engineering**, June 1992.

3- [RUMBAUGH, 1991], Jemes Rumbaugh, Michael Balaha, William Premerlani, Frederick Eddy and William Lorensen, **Object-Oriented Modeling and Design**, Englewood Cliffs, New Jersey, Prentice-Hall, Inc., 1991.

4- [BLAHA, 1988], Michael R. Blaha, William J. Rpemerlani and James E. Rumbaugh, 'Relational Database Design Using an Object-Oriented Methodology', **Communications of the ACM**, April 1988, Vol. 31, Number4.

5- [D'ANDREA,1992], Albert D'Andrea, UnisQl, Jnc., Austin, TX., 'More on Relational Database Systems', **Communications of the ACM**, Vol. 35, Number 8, August 1992

6- [GRHAM, 1991], Ian Grham, BIS Applied Systems,**Object-Oriented Models**, Addison-Wesley Publishing Company Inc., 1991.

المراجع :

1- [BAKER, 1992] , Henry G. Baker. "Relational Database", **Communications of the ACM**, April 1992, Vol. 35, No. 4.

2- [Bertino 1992], Elisa Bertino, Mauuro Negri, Giuseppe Pelagatti and Lisia Sbattella. "Object-Oriented Query Languages : The Notions and the Issues", **IEEE Transaction on Knowledge and Data Engineering**, June 1992.

3- [BLAHA, 1988], Michael R.Blaha, William J. Rpemerlani and James E. Rumbaugh, "Relational Database Design Using an Object-Oriented Methodology", **Communications of the ACM**, April 1988, Vol. 31, Number 4.

4- [BOOCH,1991], Grady Booch, **Object-Oriented Analysis and Design, with Applications**, Redwood City, Calif.: Benjamin-Cummings Publishing, Inc., 1991.

5- [Brathwaite 1991], Dr. Kenmore S. Brathwaite, **Relational Databases, Concepts, Design, and Administration**, McGraw-Hill Inc. New York,1991

6- [BROWN, 1989], A.W. BROWN, From Semantic Data Models to Object Orientation in Design Databases, **Information and Software Technology**, Vol. 31, Number 1, January/February 1989.

7- [CAMPBELL,1993], Roy H. Campbell, Nayeem Islam,David Riala and Peter Madany, "Designing and Implementing Choices: an Object-Oriented System in C++". Communications of the ACM, Vol. 36, No 9, September 1993.

8- [Castano 1998], S. Castano, V. DE Antonellis, M.G. Fugini, and B. Pernici, Conceptual Schema Analysis : Techniques and Applications, **ACM Transaction on Database Systems**, Vol. 23, No. 3, September 1998.

9- [CHU,1993], Wesly W. Chuand Ion Tim Leong, A Transaction-Based Approach to Vertical Partitioning for Relational Database Systems, **IEEE Trans. On soft.Eng.**, Vol. 19, No. 8, August 1993.

10- [Connolly 1996], Thomas. M. Connolly and Carolyn E. Begg, **Database Systems**,Addison-Wesley Publishing Co., Inc., 1996.

11- [D'ANDREA,1992], Albert D'Andrea, Unis Ql, Jnc., Austin, TX., More on Relational Database Systems, **Communications of the ACM**,Vol. 35, Number 8. August 1992.

12- [DATE,1990], C.J. Date, **An Introduction to Database Systems**, Volume I, Fifth Edition, Addison-Wesley Publishing Company Inc., 1990.

13- [DATE,1995], C.J. Date, **An Introduction to Database Systems** Volume I, Sixth Edition, Addison-Wesley Publishing Company Inc.,1995.

14- [DAVIES, 1992], P. Beynon Davies, 'Entity Models to Object Models: Object-Oriented Analysis and Database Design', **Information and Software Technology**, Vol. 34, Number 4, April 1992.

15- [ELMASRI, 1989], Ramez Elmasri and Shamkant B. Navathe, **Fundamentals of Database Systems**, Benjamin/Cummings, Redwood City, Calif., 1989.

16- [Formica 1998], A. Formica, H. D. Groger, and M. Missikoff, 'An Efficient Method Checking Object-Oriented Database Schema Correctness', **ACM Transactions on Database System**, Vol. 23, No. 3, September 1998.

17- [Fred 1991], McFaden R. Fred, Hoffer / A Jeffrey, **Database Management**. 3rd Edition, The Benjamin / Cummings Publishing Co., Inc., 1991.

18- [Fred 1994], McFaden R. Fred and Hoffer A. Jeffrey, **Modern Database Management**, 4th Edition, the Benjamin / Cummings Publishing Co., Inc., 1994

19- [GILLENSON, 1987], Mark L. Gillenson, 'The Duality of Database Structures and Design Techniques', **Communications of the ACM**, Vol. 30, Number12; December 1987.

20- [GILLENSON, 1990], Mark L. Gillenson, 'Physical Design Equivalencies in Database Conversion', **Communications of the ACM**, Vol. 33, Number 8. August 1990.

21- [GRANT, 1987], John Grant, **Logical Introduction to Databases**, Harcourt Brace Jovanovich, Publishers and its subsidiary, Academic Press, 1987.

22- [GRHAM, 1991], Ian Grham, BIS Applied Systems, **Object-Oriented Models**, Addison-Wesley Publishing Company Inc., 1991.

23- [Hurson 1993], A. R. Hurson, Simin H. Pakzad and Jia-Bing Cheng. 'Object-Oriented Database Management Systems : Evolution and Perfomance Issues', **IEEE Computer**, February 1993.

24- [Hallock, 1998], Patrick Hallock, ' Composite Objects in Relational and Object Relational Constructors Using InfoModelle ', **Journal of Conceptual Modeling**, April,1998.

25- [Jones 1993], T. Jones and I.-Y. Song, 'Binary Imposition Rules and Ternary Decomposition', In Proc. InfoScience '93 Int. Conf. On Information Science and Technology, Seoul, Korea, Korea Information Science Society, 1993.

26- [Jones 1996], T. Jones and I.-Y. Song, 'Analysis of Binary / Ternary Combinations in Entity-Relationship Modeling, **Data and Knowledge Engineering**, 19 (1) , 1996.

27- [KHOSHAFIAN, 1993], Setrag Khoshafian, **Object-Oriented Databases**, John Wiley and Sons, Inc., 1993.

28- [KORTH, 1986], Henry F. Korth and Abraham Silberschatz, **Database System** Concepts, **International Edition**, McGraw-Hill, Inc., 1986.

29- [Kroenke, 1999], David M Kroenke, **Database Processing Fundamentals, Design, and Implementaion**. 7th Edition, Prentice Hall,1999.

30- [LENZERINI, 1986], Batini C. Lenzerini and Navathe S.B., 'Acomparative analysis of Methodologies for Database Schema Integration', **ACM comput. Surv**., 18, 4, Dec., 1986.

31- [LIEBERHERR, 1993], Karl Lieberherr and Cun Xiao, 'Formal Foundations for Object-Oriented Data Modeling', **IEEE Transactions on Knowledge and Data Engineering**, Vol. 5, No. 3, June 1993.

32- [LIEBERHERR, 1993], Korl, J. Lieberherr and Cun Xiao, 'Object-Oriented software Evaluation', **IEEE Trans. on Software Engineering**, Vol. 19, No. 4, April 1993.

33- [Mcallister 1998], Mcallister, Andrew J. and Shorpe David, 'An Approach for Decomposing N-ary Data Relationships ', **Software-Practice and Experience**, Vol 28(2), Feb.,1998.

34- [MCLEOD, 1987], Hammer M. Mcleod, 'Database Description with SDM: A Semantic Database Model', **ACM Trans. Database Syst**., 19, 3, Sept. 1987

35- [NAPS, 1986], Thomas L. Naps and Bhagat Singh, **Introduction to Data Structures with Pascal**', West Publishing Company, 1986.

36- [NAVATHE, 1992], Shamkant B. Navathe, 'Evolution of Data Modeling for Databases', **Communications of the ACM**, Vol. 35 ,No. 9, September 1992.

37- [RIAD, 1993], Mokhtar Boshra Riad and Halim Habib, 'A System for Conversion Between Hierarchic, Network, and Relational Database Models', **the Egyptian Computer Science Journal**, July 1993.

38- [RIAD, 1994], Mokhtar B. Riad and Saber Abd Allah, 'Object-Oriented Databases: Features, Capabilities, Products and Development Trends . 'The 19th International Conference for Statistics, Computer Science, Scientific and Social Applications, Cairo, 9-14 April, 1994.

39- [Rochfeld 1992], A. Rochfeld and P. Negros, 'Relationship of Relationships and Other Inter-relationship Links in E-R Model', **Data and Knowledge Engineering**, 9 (2),1992.

40- [RUMBAUGH, 1991], Jemes Rumbaugh, Michael Balaha, William Premerlani, Frederick Eddy and William Lorensen, **Object-Oriented Modeling and Design**, Englewood Cliffs, New Jersey, Prentice-Hall.Inc.,1991.

41- [SCHIFFNER,1980], Scheuermann P.Schiffner, 'Abstraction Capabilities and Invariant Properties Modeling within the Entity-Relationship Approach', in **Entity-Relationship Approach to System Analysis and Design**, P.P.S. Chen, Ed., North Holland,1980.

42- [SHAER, 1994], Sally Shlaer and Stephen J. Mellor, 'Technical Correspondence' **Research in Object-Oriented Analysis and Design Communications of the ACM**, Vol. 37, No. 1,January 1994.

43- [SHAER-MELLOR,1993], Lang, N. The Shlaer-Mellor, **Object-Oriented Analysis Rules**, Soft. Eng. No.18,1,Jan.1993.

44- [SHIPMAN, 1981], Shipman D., 'The Functional Data Model and the Data Language DAPLEX'. **ACM Trans. Database Syst.**, 6, 1 Mar., 1981.

45- [Shoval 1993], P. Shoval and Shreiber, 'Database Reverse Engineering From the Relational to the Binary Relationship Model', **Data and Knowledge Engineering**, 10 (3), 1993.

46- [Stonebraker, 1998], Michael Stonebraker, Paul Brown and Dorothy Moore, **Object-Relational DBMSs** 2nd Edition, The Morgan Kaufmann Publishers, Sept. 1998.

47- [Torlone 1999], Ricardo Torlone and Paolo Atzent, 'Efficient Database Updates with Independent Schemes', **SIAM J.Computer**, Vol 28, No. 3,1999.

48- [ULLMAN,1988], Ullman, J.D., Principles of Database and Knowledge-Base System', Vol. 1, Computer Science Press, 1988.

49- [Wu 1999], C. Thomas Wu, '**An Introduction to Object-Oriented Programming with Java**', McGraw-Hill Inc. New York,1999.

المؤلف فى السطور :

الاسم :

حليم حبيب حنا سليمان ، من مواليد القاهرة ٦ يناير ١٩٥٨م .

المؤهل العلمى :

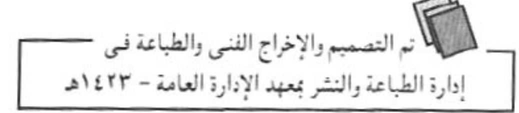
- ماجستير فى علوم الحاسب والمعلومات ١٩٩٥م ، تخصص : تصميم نظم قواعد البيانات ، جامعة القاهرة ، جمهورية مصر العربية .

العمل الحالى :

- مدير قواعد البيانات وعضو هيئة التدريس بمعهد تعليم اتحاد الفنادق الأمريكية، أورلاندو - فلوريدا - الولايات المتحدة الأمريكية .

الأنشطة العلمية :

- تصميم نظم قواعد البيانات .
- التحويل بين نظم قواعد البيانات .
- تعاونية قواعد البيانات (الذكاء الاصطناعى) .


تم التصميم والإخراج الفنى والطباعة فى
إدارة الطباعة والنشر بمعهد الإدارة العامة - ١٤٢٣هـ

هذا الكتاب

لقد تضمنت فصول هذا الكتاب العديد من الموضوعات العلمية والأسس الفنية المتعلقة بنماذج قواعد البيانات المختلفة. وتعد هذه النماذج بمنزلة الهياكل الأساسية التى تبنى عليها تقنيات قواعد البيانات. وقد ارتكز هذا الإنتاج العلمى على المحاور الرئيسية التالية:

- مفاهيم قواعد البيانات الأساسية ونماذجها المختلفة.
- تصنيف نماذج قواعد البيانات إلى كل من النماذج التقليدية والنموذج الشيئى الموجه. وتنقسم النماذج التقليدية بدورها إلى كل من النماذج التبحرية (الهرمية - الشبكية) من جهة، والنموذج العلاقى من جهة أخرى.
- ارتكاز النموذج العلاقى على الأساس العلمى لنظرية الفئات الرياضية، ومن ثم على عوامل المعالجة المرتبطة بها من خلال استخدام الجبر العلاقى والحساب العلاقى.
- دراسة وتحليل تبعية البيانات للتمييز بين قاعدة البيانات جيدة التصميم وقاعدة البيانات رديئة التصميم. وكذلك دراسة الطرق العلمية لتصميم قواعد البيانات العلاقية.
- تفاعل نماذج البيانات الدلالية فى كل أنواع نماذج قواعد البيانات.
- احتفاظ نموذج البيانات الشيئى الموجه بمناظرة مباشرة بين أشياء العالم الخارجى وقواعد البيانات؛ مما أدى إلى تحسينات معنوية لبعض التطبيقات التى يصعب معها استخدام النماذج التقليدية، مثل: الوسائط المتعددة والهندسية، وعلم الوراثة وغيرها من تلك النماذج.
- مدخل مبسط لكل من لغة الاستعلام البنائية التى ترتكز على النموذج العلاقى، ولغة الأوبال التى تعتمد على نظم قواعد البيانات الشيئية الموجهة «جيم إستون».
- الموضوعات ذات الصلة بالتقنيات المستقبلية للنظم الشيئية الموجهة مثل: الشىء العلاقى Object-Relational، ولغة الاستعلام البنائية ٣ SQL3، ومجموعة إدارة قواعد البيانات الشيئية ODMG.
- الأسس الفنية للتحويل بين النماذج الدلالية والنماذج التقليدية.
- الأسس العلمية للتحويل فيما بين النماذج التقليدية.
- قواعد التحويل من النموذج الشيئى الموجه إلى النموذج العلاقى، والاتجاه نحو قواعد البيانات الشيئية الموجهة، والأدوات الخاصة بتحليل وتصميم الشىء الموجه.

ردمك: ٥-٠٨٦-١٤.-٩٩٦٠

تصميم وإخراج وطباعة

الإدارة العامة للطباعة والنشر - معهد الإدارة العامة ١٤٢٣هـ